NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

LAUNCH DETECTION SATELLITE SYSTEM ENGINEERING ERROR ANALYSIS

by

Martin Ronald Beaulieu

March 1996

Thesis Advisor: **K. T. Alfriend**
Co-Advisor: **Thomas Jerardi**

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 1996	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE LAUNCH DETECTION SATELLITE SYSTEM ENGINEERING ERROR ANALYSIS	5. FUNDING NUMBERS
6. AUTHOR(S) Beaulieu, Martin Ronald	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

An orbiting detector of infrared (IR) energy may be used to detect the rocket plumes generated by ballistic missiles during the powered segment of their trajectory. By measuring angular directions of the detections over several observations, the trajectory properties, launch location, and impact area may be estimated using a nonlinear least-squares iteration procedure. Observations from two or more sensors may be combined to form stereoscopic lines-of-sight (LOS), increasing the accuracy of the estimation algorithm. The focus of this research has been to develop a computer-model of an estimation algorithm, and determine what parameter, or combination of parameters will significantly affect on the error of the tactical parameter estimation. This model is coded in MATLAB, and generates observation data, and produces an estimate for time, position, and heading at launch, at burnout, and calculates an impact time and position. The effects of time errors, LOS measurement errors, and satellite position errors upon the estimation accuracy were then determined using analytical and Monte Carlo simulation techniques.

14. SUBJECT TERMS DSP, Defense Support Program, TALON SHIELD / ALERT, Tactical Ballistic Missile Defense, satellite, error analysis, simulation			15. NUMBER OF PAGES 125 16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev.2-89)

Prescribed by ANSI Std. 239-18 298-102

Approved for public release; distribution is unlimited.

LAUNCH DETECTION SATELLITE SYSTEM ENGINEERING ERROR ANALYSIS

Martin R. Beaulieu
Lieutenant, United States Navy
B.S., United States Naval Academy, 1988

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ASTRONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 1996

ABSTRACT

An orbiting detector of infrared (IR) energy may be used to detect the rocket plumes generated by ballistic missiles during the powered segment of their trajectory. By measuring angular directions of the detections over several observations, the trajectory properties, launch location, and impact area may be estimated using a nonlinear least-squares iteration procedure. Observations from two or more sensors may be combined to form stereoscopic lines-of-sight (LOS), increasing the accuracy of the estimation algorithm. The focus of this research has been to develop a computer-model of an estimation algorithm, and determine what parameter, or combination of parameters will significantly affect on the error of the tactical parameter estimation. This model is coded in MATLAB, and generates observation data, and produces an estimate for time, position, and heading at launch, at burnout, and calculates an impact time and position. The effects of time errors, LOS measurement errors, and satellite position errors upon the estimation accuracy were then determined using analytical and Monte Carlo simulation techniques.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS, ACRONYMS, AND SYMBOLS

0 (zero)	subscript -- launch
A	matrix of partial derivatives of focal plane coordinates with respect to tactical parameters
a	orbit semi-major axis
α	TBM trajectory azimuth (heading) measured from true north
$a_{0..4}$	coefficients for range component polynomial of a TBM profile
ABM	anti-ballistic missile
AFSCN	Air Force Satellite Control Network
alt	altitude of TBM
AOA	angle of arrival
β	polar focal plane coordinate
$b_{0..4}$	coefficients for altitude component polynomial of a TBM profile
bo	subscript -- burnout
χ	earth-central angle between satellite and TBM
d	range component of a TBM profile including loft
δ	satellite declination
DOD	Department of Defense
DRC	Data Reduction Center
DSP	Defense Support Program
E	East component of satellite local vertical coordinate frame orbit eccentric anomaly
e	orbit eccentricity
$\hat{\mathbf{e}}$	unit line of sight direction vector in satellite local vertical coordinate frame
$\bar{\mathbf{e}}$	unit line of sight direction vector in (XYZ) coordinate frame
ϕ	geodetic latitude
f	WGS-84 flattening of the earth
ϕ'	geocentric latitude
ff	subscript -- free-flight
FOV	field of view

FPVT	focal plane vector table
γ	flight path angle
gha	Greenwich hour angle
h	altitude above WGS-84 ellipsoid
h	altitude component of a TBM profile including loft
η	polar focal plane coordinate
I	radiant intensity
ICBM	intercontinental ballistic missile
im	subscript -- impact
IR	infrared
j	iteration index
k	running index of observations
L	loft of TBM trajectory
λ	longitude
LOS	line of sight
μ	earth's gravitational parameter (398601.2 km^3/s^2)
N	North component of satellite local vertical coordinate frame
n	total number of observations
P	subscript -- nominal profile
θ	earth-central angle
ρ	distance between satellite and TBM
R	satellite position in (XYZ) coordinate frame
R	satellite radial coordinate
r	TBM position in (XYZ) coordinate frame
r_e	equatorial radius of earth (6378.137 km)
re	subscript -- reentry
r_{eff}	average radius of earth (6371 km)
r_{local}	local radius of earth
S	subscript -- satellite
S/C	spacecraft
T	alone -- time of day subscript -- TBM

t	time from launch
TBM	tactical ballistic missile
TBMD	Tactical Ballistic Missile Defense
T_{last}	time of last observation
t_{max}	maximum burn time of TBM
T_{max}	time of day of TBM burnout
T_{next}	time of next potential observation
u	Cartesian focal plane coordinate (actual observation)
$\hat{u}$	Cartesian focal plane coordinate (theoretical observation)
U	Up component of satellite local vertical coordinate frame
UEN	satellite local vertical coordinate frame (Up, East, North)
v	Cartesian focal plane coordinate (actual observation)
$\hat{v}$	Cartesian focal plane coordinate (theoretical observation)
V	velocity (speed)
WGS-84	World Geodetic Survey 1984
x	general Cartesian coordinate
XYZ	earth-centered rotating coordinate frame
y	general Cartesian coordinate
z	general Cartesian coordinate

ACKNOWLEDGMENT

Thanks to Dr. Terry Alfriend for your advice and oversight of this research. Thanks for pointing me in the right direction

Grateful thanks to Tom Jerardi for your technical expertise and acumen. Your guidance and support were invaluable.

Loving appreciation to my wife, Dr. Sandi Scrivener, who aided me in countless ways. Thank you for seeing me through.

I. INTRODUCTION

The TRW-built Defense Support Program (DSP) satellites have been the spaceborne segment of NORAD's Tactical Warning and Attack Assessment system since the early 1970's. Using infrared detectors that sense the heat from missile plumes against the Earth background, these orbiting sentries detect ballistic missile launches. DSP also detects nuclear detonations in support of Nuclear Test Ban monitoring. The DSP system provides near real-time detection information in support of DOD's tactical warning and attack assessment mission and is supported by a network of fixed and mobile ground stations that process and disseminate information to military commanders worldwide. The Cold War mission of the DSP system was to detect massive intercontinental ballistic missile (ICBM) attacks. United States' response to such an attack only required timely and unambiguous warning. Missile flight times were much longer than the time required to launch a retaliatory attack. Precise radar tracks could be established with enough time to mitigate effects as much as technology allowed.

The current combat environment demands much more from launch detection satellites. The present threat is from tactical ballistic missiles (TBM's), which exhibit much cooler burn and shorter thrust times, and possibly more depressed trajectories than those exhibited by ICBM's. TBM's can be launched from almost anywhere within a large geographical area of interest, with lofted or depressed trajectories. There may be no other sensors to quantify impact parameters in time to employ anti-ballistic missile (ABM) weapons or alert potential victims within the impact zone.

For budgetary reasons, the United States is forced to use existing launch detection satellites to counter the TBM threat into the beginning of the next century. [Ref. 1] More rapid extraction of more precise information from these existing, technology-limited satellites must be accomplished in the interim of acquiring the next-generation TBM detection satellite system. Anti-ballistic missile (ABM) systems, such as Patriot or Aegis, may be employed as ABM umbrellas in a tactical area of interest, provided they receive precise and timely cueing from these detection systems.

This raises the question: "How accurate is the information provided by DSP?" To begin to answer this question, an engineering error analysis must be performed to determine:

- What are the errors present in the detection system?
- Which errors have the greatest effect upon the accuracy of DSP output information?
- What are the effects of the errors on the results (individually and collectively)?

By modeling the algorithm used by the tactical warning system to determine trajectory elements and predict impact zone location of a detected TBM, it is then possible to introduce the inherent errors separately into the model, and then study their effects upon the results. The relative magnitudes of their effects upon the output and their effects upon the results are then determined. This is done both qualitatively and numerically (statistically), and the results are compared.

II. BACKGROUND

The DSP system consists of one or more satellites in geosynchronous orbit and one or more ground receiving stations. A DSP satellite consists of a bus (spacecraft) segment and a payload (sensor) segment. The satellite is 10 meters long, 7 meters in diameter, and weighs over 2300 kilograms. [Ref. 2]

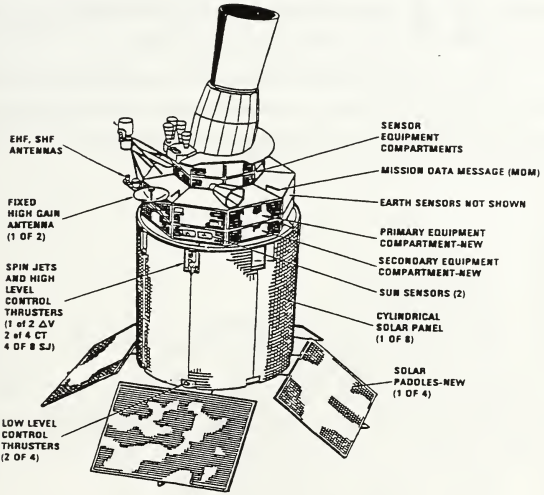


Figure 1. Diagram of DSP Satellite [From Ref. 2]

The spacecraft segment receives and transmits all commands and data, provides attitude, spin-rate, and station-keeping functions, and provides and distributes satellite power. Solar cells mounted on the spacecraft's cylindrical body and on four solar panels mounted opposite the sensor generate the electrical power. The sensor collects IR energy, provides onboard (initial) data processing, and supplies precise orientation data.

The satellite is placed in a circular, equatorial, geosynchronous orbit (Titan/IUS), oriented so that the telescope is pointed toward the earth, and spun along its longitudinal axis at six rpm. This configuration is called a "yaw spinner". The axis of rotation of the satellite is normal to the earth's surface (nadir-pointing). The satellite's spin allows the sensor to regularly scan the earth and distributes the thermal load uniformly. The spin also allows one set of attitude control components to effect two-axis earth pointing control (roll and pitch) by time-sharing. A counter-rotating momentum wheel keeps the net spin momentum near zero, thereby minimizing the gyroscopic effect due to coupling between the spin motion and the orbit rate with minimum fuel expenditure.

The IR telescope is tilted from the spin axis, so that the photo-electric cell (PEC) array covers the radius of the earth. As the satellite rotates, the entire surface of the earth within the FOV is scanned by the IR detector, shown in Figures 2 and 3.

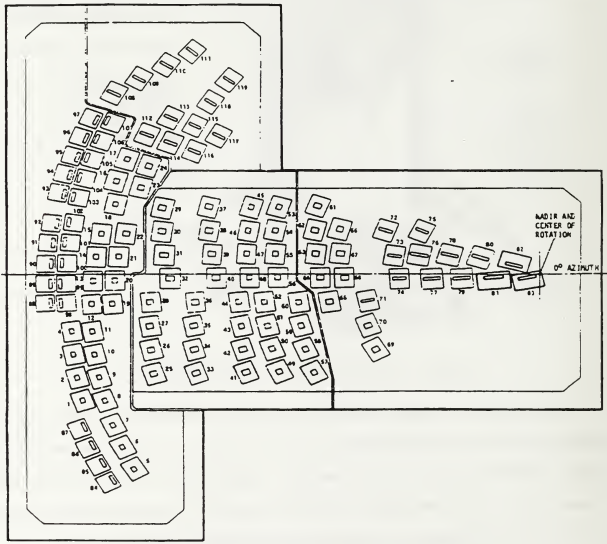


Figure 2. Photoelectric Cell Array [From Ref 2]

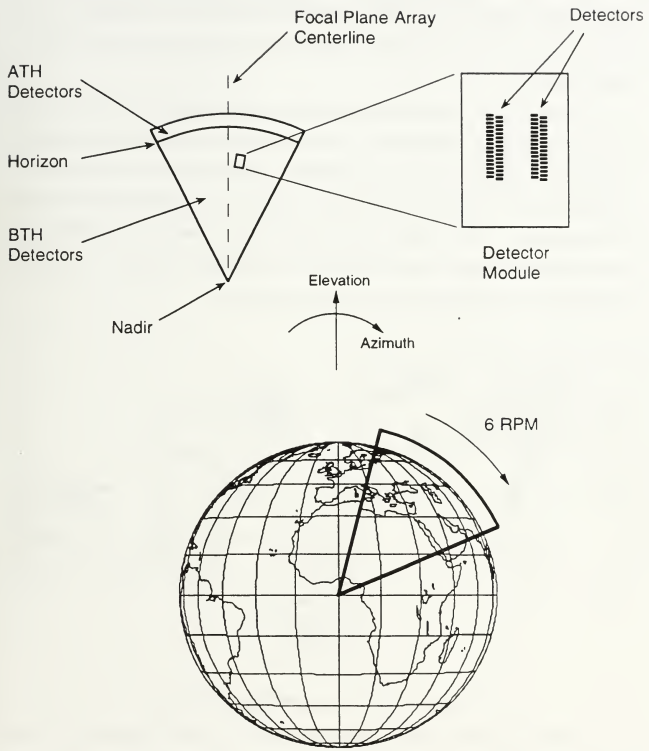


Figure 3. Focal Plane Array Scanning FOV [From Ref. 3]

Detection of IR sources is accomplished with the telescope and PEC-array portions of the sensor. The signal electronics onboard the satellite partially discriminate the targets from the background and the ground station completes the task. Location of the IR sources is derived from orientation of the IR telescope line-of-sight relative to the earth's surface. Star sensors augment the sensor data for precise determination of sensor pointing direction.

The PEC-array of the IR detector is mounted with the nadir end at the center of rotation of the telescope (see Figure 3). This array contains over 6000 detector cells that are sensitive to energy in the infrared wavelengths. As the PEC array scans the FOV, a cell passing across an IR source will generate a voltage with an amplitude proportional to the signal intensity. This voltage signal is termed an IR return, and is transmitted to ground processing stations after amplification and background filtering. A simplified diagram of the data distribution network is shown in Figure 4.

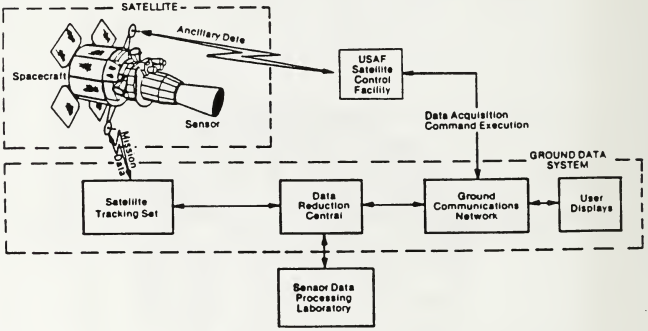


Figure 4. Data Distribution Diagram [Ref. 4]

DSP has ground stations globally positioned to receive, process, and report mission data to users. The Satellite Control Facility tracks the satellite, transmits commands, receives mission and satellite status data, and forwards the data to the Data

Reduction Center (DRC) for further processing. The DRC processes the data, extracts significant returns associated with missile launches and nuclear detonations, and generates event messages for transmission to the users over the Ground Communications Network. [Ref. 2]

III. THE ALGORITHM

{*Thomas Jerardi has graciously given his permission to adapt and modify his unpublished paper, "TALON SHIELD/ALERT State Vector Estimation", which is the basis for the majority of this chapter.*}[Ref. 5]

The detection of a TBM launch is the starting point for any ballistic missile defense. DSP does this by detecting the IR radiation emitted by the exhaust plume of a launching missile. With detections by two or more spacecraft (stereo observations), triangulation of lines of sight can be used to more accurately estimate the boost-phase trajectory, which is then used to calculate launch position, state vector (position and velocity) at missile engine burnout, and impact position. Real-time knowledge of the launch position allows targeting of the launcher. Cueing for ABM weapons systems, such as Patriot or Aegis, requires timely and accurate trajectory information, which can be propagated from knowledge of the state vector at burnout. Impact time and position data is extrapolated from the state vector at burnout, and may be used for warning personnel within the target area. Therefore, the quality (accuracy) of the trajectory estimation process is of paramount importance to Tactical Ballistic Missile Defense (TBMD). Understanding the algorithms and equations employed in the estimation process is necessary to assess the quality of the estimated parameters.

The tactical parameter estimation process is composed of several tasks: initial estimate of the tactical parameters, nonlinear least-squares estimation (refinement) of tactical parameters, burn-out time estimation, state vector generation, and impact point calculation. The tactical parameters are:

- T_0 = Time of launch
- L = Loft
- ϕ_0 = Launch point geodetic latitude
- λ_0 = Launch point geodetic longitude
- h_0 = Launch point height above WGS-84 ellipsoid
- α_0 = Flight trajectory azimuth (true heading)

These quantities will be explained in more detail in the following sections.

Observational data are used to calculate initial estimates of tactical parameters and to choose the appropriate TBM profile from a database. The profile trajectory is then used to calculate expected observations, which are then compared to the actual observations. The differences are used to calculate changes to the initial estimates, and a least-squares iteration is run until the differences between expected and actual observations are sufficiently small. The result is an accurate determination of the TBM's launch and trajectory parameters. An estimate of burnout time is then made, and the calculated trajectory is extrapolated to generate the state vector at burnout. This allows the computation of impact time and position using simple ballistic trajectory equations. Each of these tasks is described in detail in the following sections.

A. OBSERVATIONAL DATA

The starting point is the observational data, which are measurements of IR radiant intensities (IR returns) as a function of time from each of the approximately 6000 focal plane elements in the DSP satellite's sensor. Attitude information (star sensor measurements, jet-firing data, momentum wheel tachometer data) is also included in the telemetry stream and used to determine spacecraft attitude history. The satellite's position is determined by the Air Force Satellite Control Network (AFSCN). A global perspective of the observation geometry is shown in Figure 5.

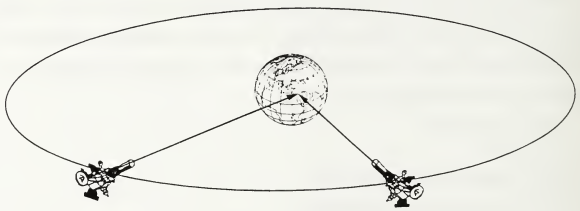


Figure 5. Observation Geometry [From Ref. 5]

The discrimination of TBM IR returns from earth background is a complex process. It will be assumed to have been effectively done, and that a table of observations corresponding to a single, boosting TBM is available. An example is shown in Table 1.

Index	Time (sec)	S/C ID	Intens.	Azimuth (rad)	Elevation (rad)	G.H.A. (rad)	Dec. (rad)	Radius (km)
k	T	S/C	I	β	η	gha	δ	R
1	129.36	1	14.0	4.061854	0.115847	0.174532	0	42164.17
2	130.30	2	23.0	2.427020	0.094741	1.221730	0	42164.17
3	135.44	3	32.0	2.062181	0.142310	1.832595	0	42164.17
4	139.36	1	29.0	4.062497	0.115971	0.174532	0	42164.17
5	140.30	2	29.0	2.427264	0.094753	1.221730	0	42164.17
6	145.44	3	40.0	2.061969	0.142405	1.832595	0	42164.17
7	149.36	1	49.0	4.063552	0.116147	0.174532	0	42164.17
8	150.30	2	60.0	2.427660	0.094746	1.221730	0	42164.17
9	155.44	3	65.0	2.061752	0.142493	1.832595	0	42164.17

Table 1. Example Observational Data

The index, k, runs from 1 to n, the total number of observations (typically less than 20), and is used as a subscript for the remaining symbols to denote a particular observation. Time, T_k, is the time of observation measured from midnight of the day of the observation, and runs between 0 and 86400 seconds. Spacecraft Identification, S/C ID, is simply a notation used to identify which satellite is making which observation. The intensity, I_k, is the radiant intensity of the IR return, and is used to select the type of TBM being detected. Azimuth angle, β_k, is the azimuth of the line of sight of the IR return measured clockwise from true south, and has a range of 0 to 2π radians (0° to 360°). Elevation angle, η_k, is the elevation of the return measured from nadir, and has values between 0 and 0.175 radians (0° to 10°). Satellite position is given in spherical coordinates. Greenwich hour angle, gha_k, is the angle between the satellite's nadir point and the prime meridian, measured positive east. Declination angle, δ_k, is the angle above or below the equator, measured positive north. (Greenwich hour angle and declination have been adopted instead of sub-satellite point longitude and latitude to avoid confusion with TBM latitude and longitude.) Radius, R_k, is the distance from the satellite to the earth's center, measured in kilometers. Geosynchronous satellites typically stay within a few degrees of the equator, and have a radius of approximately 42,164.17 km.

B. TBM PROFILE

A TBM profile is a description of the nominal powered flight trajectory of the given TBM. An example TBM profile is given in Table 2.

Time (sec)	Intensity	Altitude (km)	Range (km)	Time (sec)	Intensity	Altitude (km)	Range (km)
0	36.0	0.000	0.000	32	60.6	7.023	3.195
1	36.3	0.006	0.000	33	62.4	7.469	3.491
2	36.6	0.026	0.000	34	64.2	7.928	3.803
3	36.9	0.058	0.000	35	66.0	8.402	4.132
4	37.2	0.103	0.000	36	68.4	8.890	4.479
5	37.5	0.163	0.001	37	70.8	9.393	4.844
6	37.8	0.235	0.004	38	73.2	9.911	5.229
7	38.1	0.322	0.010	39	75.6	10.444	5.633
8	38.4	0.423	0.020	40	78.0	10.992	6.057
9	38.7	0.537	0.036	41	81.2	11.556	6.502
10	39.0	0.666	0.058	42	84.4	12.136	6.969
11	39.5	0.809	0.087	43	87.6	12.732	7.459
12	40.0	0.965	0.124	44	90.8	13.345	7.973
13	40.5	1.136	0.171	45	94.0	13.975	8.511
14	41.0	1.321	0.226	46	96.0	14.622	9.075
15	41.5	1.520	0.292	47	98.0	15.288	9.665
16	42.0	1.733	0.367	48	100.0	15.972	10.282
17	42.5	1.962	0.453	49	102.0	16.675	10.928
18	43.0	2.204	0.550	50	104.0	17.397	11.604
19	43.5	2.460	0.658	51	104.6	18.140	12.309
20	44.0	2.731	0.777	52	105.2	18.904	13.045
21	45.0	3.015	0.908	53	105.8	19.690	13.813
22	46.0	3.312	1.050	54	106.4	20.499	14.613
23	47.0	3.623	1.205	55	107.0	21.332	15.446
24	48.0	3.948	1.372	56	106.4	22.190	16.314
25	49.0	4.286	1.551	57	105.8	23.075	1.217
26	50.6	4.637	1.744	58	105.2	23.986	18.155
27	52.2	5.001	1.950	59	104.6	24.925	19.131
28	53.8	5.378	2.170	60	104.0	25.894	20.145
29	55.4	5.769	2.404	61	98.0	26.894	21.199
30	57.0	6.174	2.652	62	80.0	27.925	22.293
31	58.8	6.591	2.916	62.5	20.0	28.450	22.850

Table 2. Sample TBM Profile [Ref. 5]

A profile consists of IR intensity as a function of time, nominal (maximum range trajectory) vertical and horizontal ranges from the launch point as functions of time, and maximum burn time, t_{max} (62.5 seconds in the example). The detected radiant intensity over time is compared to TBM profiles in a data base, and the best match is selected as the type of TBM being observed. This selection process, called "typing", is complex, and is also assumed to have been done correctly. A more computationally efficient (but equivalent) representation of the nominal downrange distance and altitude profiles can be found by fitting quartic polynomials to the data in the profile tables. Use of the polynomial representation during calculations precludes table look-up and interpolation problems for non-integer times-of-flight. The profile distance polynomial has the form:

$$d_P = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 \quad (1)$$

and the profile height polynomial is:

$$h_P = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 \quad (2)$$

where t is time of flight, and $a_{1..4}$ and $b_{1..4}$ are coefficients of the fourth-order polynomial. These coefficients are not computed within this algorithm, but are assumed to be known, exact, correct, and available from the typing process; i.e., perfect *a priori* knowledge of the particular observed TBM's nominal trajectory. Since real-life TBM's do not fly the profile exactly, using the profile to determine the trajectory introduces an error into the algorithm. The inherent error of inexact profile information is ignored in this analysis. Assuming the profile to be exact does not invalidate the error analysis, however.

A "loft" parameter, L, is used to account for trajectories above or below the nominal profile. This very simple model for a ballistic missile trajectory adjusts the profile to give actual range:

$$d = (1 - 1.5L)d_P \quad (3)$$

and altitude:

$$h = (1 + L)h_P \quad (4)$$

Loft varies over approximately ±0.25. Figure 6 is a plot of nominal (L = 0), lofted (L = +0.25) and depressed (L = -0.25) trajectories.

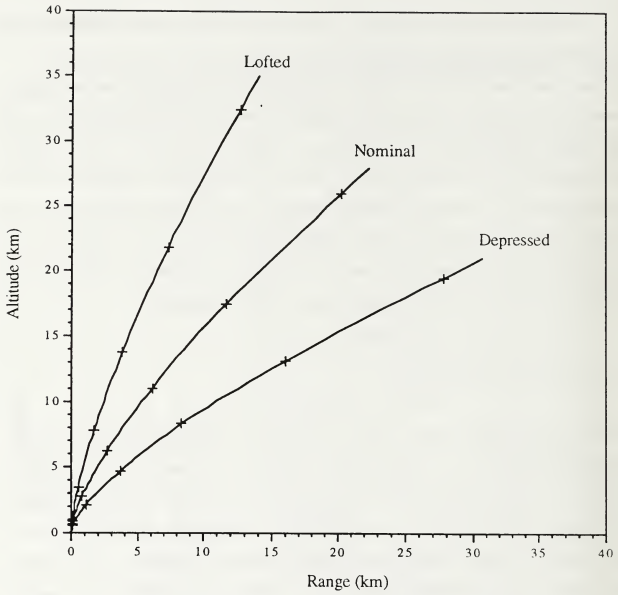


Figure 6. Trajectory Profiles for Three Loft Values

C. LINE OF SIGHT PROJECTION

The least-squares iteration process begins with an initial estimate of the tactical parameters. In order to make an estimate for the initial TBM position (λ_0, ϕ_0), the focal plane measurements must be changed into a LOS between the satellites and the TBM, which points to the position of the TBM on the globe. The first step is to transform the satellite positions (gha, δ, R) into earth-centered rotating coordinates (XYZ), in which the X-axis extends through the prime meridian and the Z axis is aligned with earth's spin axis, depicted in Figure 7.

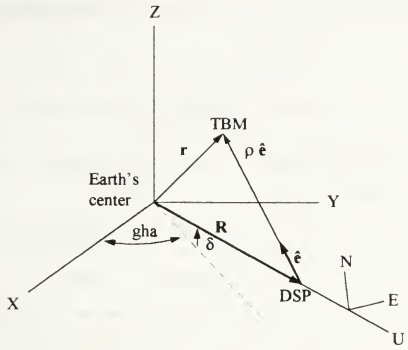


Figure 7. Coordinate Frame Illustration

The satellite position transformation from (gha, δ, R) to (XYZ) is:

$$\begin{pmatrix} x_{s_k} \\ y_{s_k} \\ z_{s_k} \end{pmatrix} = \begin{pmatrix} R_k \cos(\delta_k)\cos(gha_k) \\ R_k \cos(\delta_k)\sin(gha_k) \\ R_k \sin(\delta_k) \end{pmatrix} \tag{5}$$

where, the subscript, s, denotes satellite position, and the sub-subscript refers to the k^{th} observation. The focal plane coordinates, η_k and β_k, are first transformed to the satellite's Up-East-North reference frame, (UEN), and then must also be transformed into (XYZ). The focal plane coordinates, can be visualized with Figure 8.

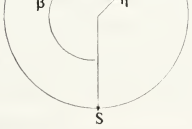


Figure 8. Focal Plane Coordinates

The transformation of polar (η_k, β_k) coordinates into Cartesian (UEN) coordinates is:

$$\bar{\mathbf{e}}_k = \begin{bmatrix} -\cos(\eta_k) \\ -\sin(\beta_k)\sin(\eta_k) \\ -\cos(\beta_k)\sin(\eta_k) \end{bmatrix} \tag{6}$$

This is the unit line-of-sight (LOS) direction vector in (UEN) coordinates. Next transform into (XYZ) (refer to Figure 7):

$$\hat{\mathbf{e}}_k = \begin{bmatrix} \cos(gha_k) & -\sin(gha_k) & 0 \\ \sin(gha_k) & \cos(gha_k) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\delta_k) & 0 & -\sin(\delta_k) \\ 0 & 1 & 0 \\ \sin(\delta_k) & 0 & \cos(\delta_k) \end{bmatrix} \bar{\mathbf{e}}_k \tag{7}$$

For the initial position estimate, it is sufficient to assume that the line of sight terminates on the surface of a spherical earth with an effective radius of r_{eff} = 6371 km. The Cartesian coordinates of the TBM are:

$$\mathbf{r}_k = \begin{bmatrix} x_{T_k} \\ y_{T_k} \\ z_{T_k} \end{bmatrix} = \mathbf{R}_k + \rho_k \hat{\mathbf{e}}_k \tag{8}$$

where the subscript, $_T$, denotes TBM position, sub-subscript k refers to the k^{th} observation, and ρ is the length of the LOS vector as shown in Figure 7. The Law of Sines is one of several methods that may be used to calculate ρ. The geometry of the problem is shown in Figure 9.

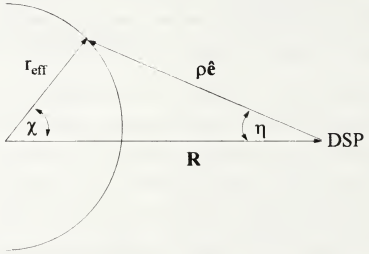


Figure 9. Initial Position Estimate Geometry

The angle opposite ρ is:

$$\chi_k = \pi - \eta_k - \sin^{-1}\left(\frac{|\mathbf{R}_k|\sin(\eta_k)}{r_{eff}}\right) \tag{9}$$

With some trigonometric identities and algebra, the expression for ρ becomes:

$$\rho_k = \frac{r_{eff}\sin(\chi_k)}{\sin(\eta_k)} = \frac{r_{eff}\sin\left[\eta_k + \sin^{-1}\left(\frac{|\mathbf{R}_k|\sin(\eta_k)}{r_{eff}}\right)\right]}{\sin(\eta_k)} \tag{10}$$

The longitude of this point is:

$$\lambda_k = \tan^{-1}\left(\frac{y_{\bar{r}_k}}{x_{\bar{r}_k}}\right) \tag{11}$$

and the latitude is:

$$\phi_k = \tan^{-1}\left(\frac{z_{\bar{r}_k}}{\sqrt{x_{\bar{r}_k}^2 + y_{\bar{r}_k}^2}}\right) \tag{12}$$

D. INITIAL ESTIMATES

The core of the estimation process is a nonlinear least-squares iterative calculation of the tactical parameters. This method is based on a one-term (linear) Taylor expansion of the focal plane observations in terms of the six tactical parameters. This procedure requires an initial estimate of the tactical parameters. The accuracy of the initial estimate does not effect the accuracy of the final estimate, but can effect the convergence time of the least-squares process.

The initial estimate of the launch position and heading is based on the projected positions (ϕ_k and λ_k) obtained from the LOS observations. The lines-of-sight from each satellite usually do not intersect at a single point (even if they were simultaneous observations), as shown in Figure 10. The initial latitude and longitude guesses are found by calculating the average position from the first LOS observation from each satellite. The assumption is made that if one satellite can see the TBM, then all can observe it. This artificiality is not true in the physical world.

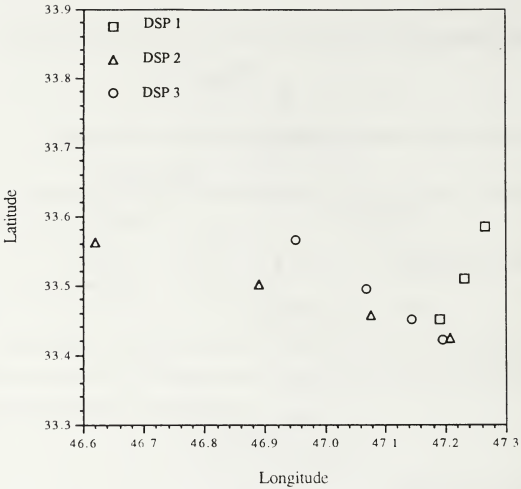


Figure 10. LOS Projection Scatter

In this example, three satellites observe the TBM, so the first LOS projected latitudes and longitudes (from each satellite) are averaged:

$$\phi_0^{(0)} = \frac{\phi_1 + \phi_2 + \phi_3}{3} \tag{13}$$

$$\lambda_0^{(0)} = \frac{\lambda_1 + \lambda_2 + \lambda_3}{3} \tag{14}$$

The superscript, $^{(0)}$, denotes the initial estimate. The last LOS projected positions are also averaged to obtain the approximate final position observed:

$$\phi_{last} = \frac{\phi_{n-2} + \phi_{n-1} + \phi_n}{3} \tag{15}$$

$$\lambda_{last} = \frac{\lambda_{n-2} + \lambda_{n-1} + \lambda_n}{3} \tag{16}$$

Equations (13) to (16) assume that the first and last three observations are from three different satellites. If only two satellites observe the launch, only the first and last two observations would be averaged.

The direction of the last position from the first is the initial estimate for the heading, $\alpha_0^{(0)}$:

$$\alpha_0^{(0)} = \frac{\pi}{2} - \tan_2^{-1}[(\lambda_{last} - \lambda_0^{(0)})\cos(\phi_0^{(0)}), \phi_{last} - \phi_0^{(0)}] \tag{17}$$

The function $\tan_2^{-1}$ is the two-argument arctangent, used here because its range is $\pm\pi$. This function is used because a four-quadrant answer is required.

The first estimate for T_0 is twenty seconds before the first observation:

time of launch

$$T_0^{(0)} = T_1 - 20 \tag{18}$$

and the initial guess for L and h_0 are both zero:

$$L^{(0)} = 0 \tag{19}$$

$$h_0^{(0)} = 0 \tag{20}$$

E. EXPECTED POSITIONS ALONG THE TRAJECTORY

Given the six (estimated) tactical parameters at launch, the expected position of the TBM along its boost trajectory can be determined by applying the TBM profile to any observation time, and in particular, T_k, by first computing:

$$t_k = T_k - T_0 \tag{21}$$

Equations (1) through (4) then give the expected altitude (h_k) and range (d_k) from the estimated launch point when $t = t_k$:

$$d_{P_k} = a_0 + a_1 t_k + a_2 t_k^2 + a_3 t_k^3 + a_4 t_k^4 \tag{1}$$

$$h_{P_k} = b_0 + b_1 t_k + b_2 t_k^2 + b_3 t_k^3 + b_4 t_k^4 \tag{2}$$

$$d_k = (1 - 1.5L) d_{P_k} \tag{3}$$

$$h_k = (1 + L) h_{P_k} \tag{4}$$

Next, it is necessary to calculate the "earth-central angle", a quantity used in (23) and (24):

$$\theta_k = \frac{d_k}{r_{eff}} \tag{22}$$

θ_k is simply the angle between the launch point and the position of the TBM with the earth's center as the vertex. θ can be visualized with Figure 11.

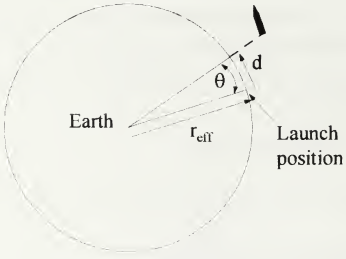


Figure 11. Earth-central Angle

Using θ_k, spherical trigonometry gives the equations for the geodetic latitude:

$$\phi_k = \frac{\pi}{2} - \cos^{-1}\left[\cos(\theta_k)\sin(\phi_0) + \sin(\theta_k)\cos(\phi_0)\cos(\alpha_0)\right] \tag{23}$$

and longitude:

$$\lambda_k = \lambda_0 + \sin^{-1}\left[\frac{\sin(\theta_k)\sin(\alpha_0)}{\cos(\phi_k)}\right] \tag{24}$$

The altitude is simply the present height of the TBM added to the initial height at launch:

$$alt_k = h_0 + h_k \tag{25}$$

In addition to the geodetic coordinates, the Cartesian coordinates are also required. Using r_e = 6378.137 km, and $f = \frac{1}{298.257}$, the WGS-84 values for the flattening of the earth, the geocentric latitude of the TBM is calculated:

$$\phi'_k = \tan^{-1}[(1-f)^2 \tan(\phi_k)] \tag{26}$$

as well as the local radius:

$$r_{local_k} = \frac{r_e(1-f')}{\sqrt{(1-f')^2\cos^2(\phi'_k) + \sin^2(\phi'_k)}} \tag{27}$$

These values are then transformed into earth-centered, Cartesian coordinates of the TBM:

$$x_{T_k} = \left[r_{local_k}\cos(\phi'_k) + alt_k\cos(\phi_k)\right]\cos(\lambda_k) \tag{28}$$

$$y_{T_k} = \left[r_{local_k}\cos(\phi'_k) + alt_k\cos(\phi_k)\right]\sin(\lambda_k) \tag{29}$$

$$z_{T_k} = r_{local_k}\sin(\phi'_k) + alt_k\sin(\phi_k) \tag{30}$$

F. NONLINEAR LEAST SQUARES ESTIMATION

Since the (polar) focal plane coordinates of the observations are nonlinear functions of the tactical parameters, a nonlinear least-squares process is required. This is achieved by a sequence of linear least-squares estimations. Each of these linear least squares estimations is based on a one-term (linear) Taylor series expansion of the observations in terms of the tactical parameters. A general requirement of ordinary linear least squares is that the noise contaminating the observations should be independent and have a common (preferably Gaussian) distribution.

In this case, the "polar" focal plane observations (β_k, η_k) are transformed into "Cartesian-like" coordinates (u_k, v_k):

$$u_k = -\tan(\eta_k)\sin(\beta_k) \tag{31}$$

$$v_k = -\tan(\eta_k)\cos(\beta_k) \tag{32}$$

so that the coordinates have similar behavior with respect to errors and noise. Either η or $\sin(\eta)$ could be used in place of $\tan(\eta)$, but since $\eta<10°$, all three behave similarly. The chosen form corresponds to a gnomonic projection of the globe onto a plane tangent at the nadir point. Figure 12 illustrates this.

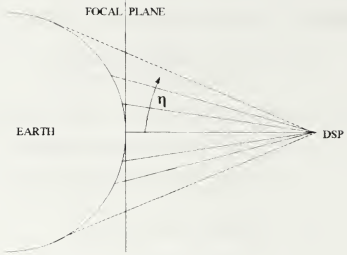


Figure 12. Focal Plane Transformation

Next, it is necessary to determine what the focal plane coordinates would be for each observation if the TBM were actually located at the predicted coordinates given in (28) to (30).

The expected line-of-sight is computed:

$$\begin{pmatrix} \Delta x_k \\ \Delta y_k \\ \Delta z_k \end{pmatrix} = \begin{pmatrix} x_{T_k} - x_{S_k} \\ y_{T_k} - y_{S_k} \\ z_{T_k} - z_{S_k} \end{pmatrix} \tag{33}$$

In order to determine the focal plane coordinates of azimuth, β, and elevation, η, the LOS vector given in (33) must be rotated into the observing satellite's local vertical coordinate frame, (UEN):

$$\begin{pmatrix} U_k \\ E_k \\ N_k \end{pmatrix} = \begin{pmatrix} \cos(\delta_k) & 0 & \sin(\delta_k) \\ 0 & 1 & 0 \\ -\sin(\delta_k) & 0 & \cos(\delta_k) \end{pmatrix} \begin{pmatrix} \cos(gha_k) & \sin(gha_k) & 0 \\ -\sin(gha_k) & \cos(gha_k) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Delta x_k \\ \Delta y_k \\ \Delta z_k \end{pmatrix} \tag{34}$$

This is the transpose of the transformation matrices given in (7).

Now $\hat{u}$ and $\hat{v}$ can be simply expressed in (UEN) terms by noting that:

$$\sin(\hat{\beta}_k) = \frac{-E_k}{\sqrt{E_k^2 + N_k^2}} \tag{35}$$

$$\cos(\hat{\beta}_k) = \frac{-N_k}{\sqrt{E_k^2 + N_k^2}} \tag{36}$$

Expected focal plane coordinates, $(\hat{u}_k, \hat{v}_k)$, are then:

$$\hat{u}_k = -\frac{E_k}{U_k} \tag{37}$$

$$\hat{v}_k = -\frac{N_k}{U_k} \tag{38}$$

Expected focal plane coordinates are denoted with a "^". Taking the difference between actual observations and expected observations generates δu and δv (found on the left-hand sides of (41) and (42)):

$$\delta u_k = u_k - \hat{u}_k \tag{39}$$

$$\delta v_k = v_k - \hat{v}_k \tag{40}$$

Now that the changes in the focal plane coordinates are known, they can be related to changes in the tactical parameters. The one-term Taylor expansions of u and v in terms of the tactical parameters are:

$$\delta u = \frac{\partial u}{\partial T_0}\delta T_0 + \frac{\partial u}{\partial L}\delta L + \frac{\partial u}{\partial \phi_0}\delta\phi_0 + \frac{\partial u}{\partial \lambda_0}\delta\lambda_0 + \frac{\partial u}{\partial h_0}\delta h_0 + \frac{\partial u}{\partial \alpha_0}\delta\alpha_0 \tag{41}$$

$$\delta v = \frac{\partial v}{\partial T_0}\delta T_0 + \frac{\partial v}{\partial L}\delta L + \frac{\partial v}{\partial \phi_0}\delta\phi_0 + \frac{\partial v}{\partial \lambda_0}\delta\lambda_0 + \frac{\partial v}{\partial h_0}\delta h_0 + \frac{\partial v}{\partial \alpha_0}\delta\alpha_0 \tag{42}$$

These equations, written in vector-matrix format, are the linear least-squares model that forms the basis of the tactical parameter estimation process:

$$\begin{pmatrix} \delta u_k \\ \delta v_k \end{pmatrix} = \begin{pmatrix} \frac{\partial u_k}{\partial T_0} & \frac{\partial u_k}{\partial L} & \frac{\partial u_k}{\partial \phi_0} & \frac{\partial u_k}{\partial \lambda_0} & \frac{\partial u_k}{\partial h_0} & \frac{\partial u_k}{\partial \alpha_0} \\ \frac{\partial v_k}{\partial T_0} & \frac{\partial v_k}{\partial L} & \frac{\partial v_k}{\partial \phi_0} & \frac{\partial v_k}{\partial \lambda_0} & \frac{\partial v_k}{\partial h_0} & \frac{\partial v_k}{\partial \alpha_0} \end{pmatrix} \begin{pmatrix} \delta T_0 \\ \delta L \\ \delta\phi_0 \\ \delta\lambda_0 \\ \delta h_0 \\ \delta\alpha_0 \end{pmatrix} \tag{43}$$

There are two rows of the center matrix in (43) for each of the k = (1 to n) observations for a total of 2n rows. This matrix of partials is denoted the "**A**" matrix, and represents changes in the focal plane coordinates with respect to changes in tactical parameters.

The differences in the focal plane coordinates, δu and δv, are used to generate adjustments to the tactical parameters by solving (43) for the tactical parameter variations:

$$\begin{pmatrix} \delta T_0 \\ \delta L \\ \delta \phi_0 \\ \delta \lambda_0 \\ \delta h_0 \\ \delta \alpha_0 \end{pmatrix} = \left[\mathbf{A}^T \mathbf{A} \right]^{-1} \mathbf{A}^T \begin{pmatrix} \delta u_k \\ \delta v_k \end{pmatrix} \tag{44}$$

The middle term in (44) is the pseudoinverse of **A**, and is used because **A** is an (2n x 6) matrix, and is generally not square. The changes to the tactical parameters, the left-hand matrix in (44), are added to the initial estimate, and the process is repeated until all the components of the left-hand vector in (44) are sufficiently small, or the convergence criterion is met. The result is the correct tactical parameters for the observed TBM launch.

The matrix of partial derivatives, **A**, is the key to the whole iteration process. The development of this matrix involves multiple use of the Chain Rule for derivatives. The partials in (41) can be expanded to yield:

$$\frac{\partial u}{\partial T_0} = \frac{\partial u}{\partial \phi}\frac{\partial \phi}{\partial T_0} + \frac{\partial u}{\partial \lambda}\frac{\partial \lambda}{\partial T_0} + \frac{\partial u}{\partial h}\frac{\partial h}{\partial T_0} \tag{45}$$

$$\frac{\partial u}{\partial L} = \frac{\partial u}{\partial \phi}\frac{\partial \phi}{\partial L} + \frac{\partial u}{\partial \lambda}\frac{\partial \lambda}{\partial L} + \frac{\partial u}{\partial h}\frac{\partial h}{\partial L} \tag{46}$$

$$\frac{\partial u}{\partial \phi_0} = \frac{\partial u}{\partial \phi}\frac{\partial \phi}{\partial \phi_0} + \frac{\partial u}{\partial \lambda}\frac{\partial \lambda}{\partial \phi_0} + \frac{\partial u}{\partial h}\frac{\partial h}{\partial \phi_0} \tag{47}$$

$$\frac{\partial u}{\partial \lambda_0} = \frac{\partial u}{\partial \phi}\frac{\partial \phi}{\partial \lambda_0} + \frac{\partial u}{\partial \lambda}\frac{\partial \lambda}{\partial \lambda_0} + \frac{\partial u}{\partial h}\frac{\partial h}{\partial \lambda_0} \tag{48}$$

$$\frac{\partial u}{\partial h_0} = \frac{\partial u}{\partial \phi}\frac{\partial \phi}{\partial h_0} + \frac{\partial u}{\partial \lambda}\frac{\partial \lambda}{\partial h_0} + \frac{\partial u}{\partial h}\frac{\partial h}{\partial h_0} \tag{49}$$

$$\frac{\partial u}{\partial \alpha_0} = \frac{\partial u}{\partial \phi}\frac{\partial \phi}{\partial \alpha_0} + \frac{\partial u}{\partial \lambda}\frac{\partial \lambda}{\partial \alpha_0} + \frac{\partial u}{\partial h}\frac{\partial h}{\partial \alpha_0} \tag{50}$$

In a similar manner, the partials of (42) are expanded:

$$\frac{\partial v}{\partial T_0} = \frac{\partial v}{\partial \phi}\frac{\partial \phi}{\partial T_0} + \frac{\partial v}{\partial \lambda}\frac{\partial \lambda}{\partial T_0} + \frac{\partial v}{\partial h}\frac{\partial h}{\partial T_0} \tag{51}$$

$$\frac{\partial v}{\partial L} = \frac{\partial v}{\partial \phi}\frac{\partial \phi}{\partial L} + \frac{\partial v}{\partial \lambda}\frac{\partial \lambda}{\partial L} + \frac{\partial v}{\partial h}\frac{\partial h}{\partial L} \tag{52}$$

$$\frac{\partial v}{\partial \phi_0} = \frac{\partial v}{\partial \phi}\frac{\partial \phi}{\partial \phi_0} + \frac{\partial v}{\partial \lambda}\frac{\partial \lambda}{\partial \phi_0} + \frac{\partial v}{\partial h}\frac{\partial h}{\partial \phi_0} \tag{53}$$

$$\frac{\partial v}{\partial \lambda_0} = \frac{\partial v}{\partial \phi}\frac{\partial \phi}{\partial \lambda_0} + \frac{\partial v}{\partial \lambda}\frac{\partial \lambda}{\partial \lambda_0} + \frac{\partial v}{\partial h}\frac{\partial h}{\partial \lambda_0} \tag{54}$$

$$\frac{\partial v}{\partial h_0} = \frac{\partial v}{\partial \phi}\frac{\partial \phi}{\partial h_0} + \frac{\partial v}{\partial \lambda}\frac{\partial \lambda}{\partial h_0} + \frac{\partial v}{\partial h}\frac{\partial h}{\partial h_0} \tag{55}$$

$$\frac{\partial v}{\partial \alpha_0} = \frac{\partial v}{\partial \phi}\frac{\partial \phi}{\partial \alpha_0} + \frac{\partial v}{\partial \lambda}\frac{\partial \lambda}{\partial \alpha_0} + \frac{\partial v}{\partial h}\frac{\partial h}{\partial \alpha_0} \tag{56}$$

Patterns and structure in these twelve equations are more easily recognized when these equations are written in matrix form:

$$\begin{pmatrix} \frac{\partial u}{\partial T_0} \\ \frac{\partial u}{\partial L} \\ \frac{\partial u}{\partial \phi_0} \\ \frac{\partial u}{\partial \lambda_0} \\ \frac{\partial u}{\partial h_0} \\ \frac{\partial u}{\partial \alpha_0} \end{pmatrix} = \begin{bmatrix} \frac{\partial \phi}{\partial T_0} & \frac{\partial \lambda}{\partial T_0} & \frac{\partial h}{\partial T_0} \\ \frac{\partial \phi}{\partial L} & \frac{\partial \lambda}{\partial L} & \frac{\partial h}{\partial L} \\ \frac{\partial \phi}{\partial \phi_0} & \frac{\partial \lambda}{\partial \phi_0} & \frac{\partial h}{\partial \phi_0} \\ \frac{\partial \phi}{\partial \lambda_0} & \frac{\partial \lambda}{\partial \lambda_0} & \frac{\partial h}{\partial \lambda_0} \\ \frac{\partial \phi}{\partial h_0} & \frac{\partial \lambda}{\partial h_0} & \frac{\partial h}{\partial h_0} \\ \frac{\partial \phi}{\partial \alpha_0} & \frac{\partial \lambda}{\partial \alpha_0} & \frac{\partial h}{\partial \alpha_0} \end{bmatrix} \begin{pmatrix} \frac{\partial u}{\partial \phi} \\ \frac{\partial u}{\partial \lambda} \\ \frac{\partial u}{\partial h} \end{pmatrix} \tag{57}$$

$$\begin{pmatrix} \frac{\partial v}{\partial T_0} \\ \frac{\partial v}{\partial L} \\ \frac{\partial v}{\partial \phi_0} \\ \frac{\partial v}{\partial \lambda_0} \\ \frac{\partial v}{\partial h_0} \\ \frac{\partial v}{\partial \alpha_0} \end{pmatrix} = \begin{bmatrix} \frac{\partial \phi}{\partial T_0} & \frac{\partial \lambda}{\partial T_0} & \frac{\partial h}{\partial T_0} \\ \frac{\partial \phi}{\partial L} & \frac{\partial \lambda}{\partial L} & \frac{\partial h}{\partial L} \\ \frac{\partial \phi}{\partial \phi_0} & \frac{\partial \lambda}{\partial \phi_0} & \frac{\partial h}{\partial \phi_0} \\ \frac{\partial \phi}{\partial \lambda_0} & \frac{\partial \lambda}{\partial \lambda_0} & \frac{\partial h}{\partial \lambda_0} \\ \frac{\partial \phi}{\partial h_0} & \frac{\partial \lambda}{\partial h_0} & \frac{\partial h}{\partial h_0} \\ \frac{\partial \phi}{\partial \alpha_0} & \frac{\partial \lambda}{\partial \alpha_0} & \frac{\partial h}{\partial \alpha_0} \end{bmatrix} \begin{pmatrix} \frac{\partial v}{\partial \phi} \\ \frac{\partial v}{\partial \lambda} \\ \frac{\partial v}{\partial h} \end{pmatrix} \tag{58}$$

Note here that (57) and (58) have identical structure. The left-hand sides are the derivatives of the focal plane coordinates with respect to the tactical parameters, the vectors on the right are the derivatives of the corresponding focal plane coordinates with respect to TBM position (latitude, longitude, and height), and the center matrix describes the derivatives of the position with respect the the tactical parameters. This effectively separates the change in the focal plane coordinates with respect to tactical parameters into two parts: a matrix that describes changes in position due to trajectory, and a vector that describes the relation between focal plane observations and TBM position.

The matrix is given a name, $\mathbf{A}_1$:

$$\mathbf{A}_1 = \begin{bmatrix} \frac{\partial \phi}{\partial T_0} & \frac{\partial \lambda}{\partial T_0} & \frac{\partial h}{\partial T_0} \\ \frac{\partial \phi}{\partial L} & \frac{\partial \lambda}{\partial L} & \frac{\partial h}{\partial L} \\ \frac{\partial \phi}{\partial \phi_0} & \frac{\partial \lambda}{\partial \phi_0} & \frac{\partial h}{\partial \phi_0} \\ \frac{\partial \phi}{\partial \lambda_0} & \frac{\partial \lambda}{\partial \lambda_0} & \frac{\partial h}{\partial \lambda_0} \\ \frac{\partial \phi}{\partial h_0} & \frac{\partial \lambda}{\partial h_0} & \frac{\partial h}{\partial h_0} \\ \frac{\partial \phi}{\partial \alpha_0} & \frac{\partial \lambda}{\partial \alpha_0} & \frac{\partial h}{\partial \alpha_0} \end{bmatrix} \tag{59}$$

so that (57) and (58) can be rewritten:

$$\begin{pmatrix} \frac{\partial u}{\partial T_0} \\ \frac{\partial u}{\partial L} \\ \frac{\partial u}{\partial \phi_0} \\ \frac{\partial u}{\partial \lambda_0} \\ \frac{\partial u}{\partial h_0} \\ \frac{\partial u}{\partial \alpha_0} \end{pmatrix} = \mathbf{A}_1 \begin{pmatrix} \frac{\partial u}{\partial \phi} \\ \frac{\partial u}{\partial \lambda} \\ \frac{\partial u}{\partial h} \end{pmatrix} \tag{60}$$

$$\begin{pmatrix} \frac{\partial v}{\partial T_0} \\ \frac{\partial v}{\partial L} \\ \frac{\partial v}{\partial \phi_0} \\ \frac{\partial v}{\partial \lambda_0} \\ \frac{\partial v}{\partial h_0} \\ \frac{\partial v}{\partial \alpha_0} \end{pmatrix} = \mathbf{A}_1 \begin{pmatrix} \frac{\partial v}{\partial \phi} \\ \frac{\partial v}{\partial \lambda} \\ \frac{\partial v}{\partial h} \end{pmatrix} \tag{61}$$

The elements of $\mathbf{A}_1$ can be found qualitatively. This is done in Appendix A, and the results are:

$$\mathbf{A}_1 = \begin{bmatrix} \frac{\partial d}{\partial T_0}\frac{\cos(\alpha_0)}{r_{eff}} & \frac{\partial d}{\partial T_0}\frac{\sin(\alpha_0)}{r_{eff}\cos(\phi_{0)}} & \frac{\partial h}{\partial T_0} \\ -\frac{1.5 d_p \cos(\alpha_0)}{r_{eff}} & -\frac{1.5 d_p \sin(\alpha_0)}{r_{eff}\cos(\phi_0)} & h_P \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\frac{d\sin(\alpha_0)}{r_{eff}} & \frac{d\cos(\alpha_0)}{r_{eff}\cos(\phi_0)} & 0 \end{bmatrix} \tag{62}$$

The values of the elements in (62) are approximations, but approximate magnitudes and correct signs are all that are required for the iteration process to converge. The benefit of making this simplification for $\mathbf{A}_1$ is the reduction in computation time. The complete and exact formulae can be extrapolated from Appendix A.

Similarly, vectors on the right-hand sides of (57) and (58) can be separated into simpler components:

$$\frac{\partial u}{\partial \phi} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \phi} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \phi} + \frac{\partial u}{\partial z}\frac{\partial z}{\partial \phi} \tag{63}$$

$$\frac{\partial u}{\partial \lambda} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \lambda} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \lambda} + \frac{\partial u}{\partial z}\frac{\partial z}{\partial \lambda} \tag{64}$$

$$\frac{\partial u}{\partial h} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial h} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial h} + \frac{\partial u}{\partial z}\frac{\partial z}{\partial h} \tag{65}$$

$$\frac{\partial v}{\partial \phi} = \frac{\partial v}{\partial x}\frac{\partial x}{\partial \phi} + \frac{\partial v}{\partial y}\frac{\partial y}{\partial \phi} + \frac{\partial v}{\partial z}\frac{\partial z}{\partial \phi} \tag{66}$$

$$\frac{\partial v}{\partial \lambda} = \frac{\partial v}{\partial x}\frac{\partial x}{\partial \lambda} + \frac{\partial v}{\partial y}\frac{\partial y}{\partial \lambda} + \frac{\partial v}{\partial z}\frac{\partial z}{\partial \lambda} \tag{67}$$

$$\frac{\partial v}{\partial h} = \frac{\partial v}{\partial x}\frac{\partial x}{\partial h} + \frac{\partial v}{\partial y}\frac{\partial y}{\partial h} + \frac{\partial v}{\partial z}\frac{\partial z}{\partial h} \tag{68}$$

As before, these equations can be written in matrix form:

$$\begin{pmatrix} \frac{\partial u}{\partial \phi} \\ \frac{\partial u}{\partial \lambda} \\ \frac{\partial u}{\partial h} \end{pmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \phi} & \frac{\partial y}{\partial \phi} & \frac{\partial z}{\partial \phi} \\ \frac{\partial x}{\partial \lambda} & \frac{\partial y}{\partial \lambda} & \frac{\partial z}{\partial \lambda} \\ \frac{\partial x}{\partial h} & \frac{\partial y}{\partial h} & \frac{\partial z}{\partial h} \end{bmatrix} \begin{pmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial u}{\partial z} \end{pmatrix} \tag{69}$$

$$\begin{pmatrix} \frac{\partial v}{\partial \phi} \\ \frac{\partial v}{\partial \lambda} \\ \frac{\partial v}{\partial h} \end{pmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \phi} & \frac{\partial y}{\partial \phi} & \frac{\partial z}{\partial \phi} \\ \frac{\partial x}{\partial \lambda} & \frac{\partial y}{\partial \lambda} & \frac{\partial z}{\partial \lambda} \\ \frac{\partial x}{\partial h} & \frac{\partial y}{\partial h} & \frac{\partial z}{\partial h} \end{bmatrix} \begin{pmatrix} \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial v}{\partial z} \end{pmatrix} \tag{70}$$

The center matrix is renamed $\mathbf{A}_2$:

$$\mathbf{A}_2 = \begin{bmatrix} \frac{\partial x}{\partial \phi} & \frac{\partial y}{\partial \phi} & \frac{\partial z}{\partial \phi} \\ \frac{\partial x}{\partial \lambda} & \frac{\partial y}{\partial \lambda} & \frac{\partial z}{\partial \lambda} \\ \frac{\partial x}{\partial h} & \frac{\partial y}{\partial h} & \frac{\partial z}{\partial h} \end{bmatrix} \tag{71}$$

and its elements can be approximated by:

$$\mathbf{A}_2 = \begin{bmatrix} -(r_{eff} + h)\sin(\phi)\cos(\lambda) & -(r_{eff} + h)\sin(\phi)\sin(\lambda) & (r_{eff} + h)\cos(\phi) \\ -(r_{eff} + h)\cos(\phi)\sin(\lambda) & (r_{eff} + h)\cos(\phi)\cos(\lambda) & 0 \\ \cos(\phi)\cos(\lambda) & \cos(\phi)\sin(\lambda) & \sin(\phi) \end{bmatrix} \tag{72}$$

The vectors on the right-hand side of (69) and (70) can be further broken down:

$$\begin{pmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial u}{\partial z} \end{pmatrix} = \begin{bmatrix} \frac{\partial U}{\partial x} & \frac{\partial E}{\partial x} & \frac{\partial N}{\partial x} \\ \frac{\partial U}{\partial y} & \frac{\partial E}{\partial y} & \frac{\partial N}{\partial y} \\ \frac{\partial U}{\partial z} & \frac{\partial E}{\partial z} & \frac{\partial N}{\partial z} \end{bmatrix} \begin{pmatrix} \frac{\partial u}{\partial U} \\ \frac{\partial u}{\partial E} \\ \frac{\partial u}{\partial N} \end{pmatrix} \tag{73}$$

$$\begin{pmatrix} \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial v}{\partial z} \end{pmatrix} = \begin{bmatrix} \frac{\partial U}{\partial x} & \frac{\partial E}{\partial x} & \frac{\partial N}{\partial x} \\ \frac{\partial U}{\partial y} & \frac{\partial E}{\partial y} & \frac{\partial N}{\partial y} \\ \frac{\partial U}{\partial z} & \frac{\partial E}{\partial z} & \frac{\partial N}{\partial z} \end{bmatrix} \begin{pmatrix} \frac{\partial v}{\partial U} \\ \frac{\partial v}{\partial E} \\ \frac{\partial v}{\partial N} \end{pmatrix} \tag{74}$$

The center matrices in (73) and (74) are identical, and renamed $\mathbf{A}_3$:

$$\mathbf{A}_3 = \begin{bmatrix} \frac{\partial U}{\partial x} & \frac{\partial E}{\partial x} & \frac{\partial N}{\partial x} \\ \frac{\partial U}{\partial y} & \frac{\partial E}{\partial y} & \frac{\partial N}{\partial y} \\ \frac{\partial U}{\partial z} & \frac{\partial E}{\partial z} & \frac{\partial N}{\partial z} \end{bmatrix} \tag{75}$$

$\mathbf{A}_3$, th transformation from (XYZ) to (UEN) has been derived previously in (7):

$$\mathbf{A}_3 = \begin{bmatrix} \cos(gha_k) & -\sin(gha_k) & 0 \\ \sin(gha_k) & \cos(gha_k) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\delta_k) & 0 & -\sin(\delta_k) \\ 0 & 1 & 0 \\ \sin(\delta_k) & 0 & \cos(\delta_k) \end{bmatrix} \tag{76}$$

The final step is to determine the elements in the vectors on the right-hand sides of (73) and (74). These are easily computed from (37) and (38):

$$\frac{\partial u}{\partial U} = \frac{E}{U^2} \tag{77}$$

$$\frac{\partial u}{\partial E} = -\frac{1}{U} \tag{78}$$

$$\frac{\partial u}{\partial N} = 0 \tag{79}$$

$$\frac{\partial v}{\partial U} = \frac{N}{U^2} \tag{80}$$

$$\frac{\partial v}{\partial E} = 0 \tag{81}$$

$$\frac{\partial v}{\partial N} = -\frac{1}{U} \tag{82}$$

Now (57) and (58) can be rewritten:

$$\begin{pmatrix} \frac{\partial u}{\partial T_0} \\ \frac{\partial u}{\partial L} \\ \frac{\partial u}{\partial \phi_0} \\ \frac{\partial u}{\partial \lambda_0} \\ \frac{\partial u}{\partial h_0} \\ \frac{\partial u}{\partial \alpha_0} \end{pmatrix} = \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3 \begin{pmatrix} \frac{E}{U^2} \\ -\frac{1}{U} \\ 0 \end{pmatrix} \tag{83}$$

$$\begin{pmatrix} \frac{\partial v}{\partial T_0} \\ \frac{\partial v}{\partial L} \\ \frac{\partial v}{\partial \phi_0} \\ \frac{\partial v}{\partial \lambda_0} \\ \frac{\partial v}{\partial h_0} \\ \frac{\partial v}{\partial \alpha_0} \end{pmatrix} = \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3 \begin{pmatrix} \frac{N}{U^2} \\ 0 \\ -\frac{1}{U} \end{pmatrix} \tag{84}$$

Equation (83) is used to compute the odd rows (k = odd) of **A** in (43), and (84) is used to compute the even rows (k = even) of **A** in (43).

Focal plane coordinates, (β_k, η_k), are transformed into Cartesian-like focal plane coordinates, (u_k, v_k) ①. (Circled numbers refer to steps illustrated in Figure 13.) Satellite positions and initial estimates of tactical parameters are used to generate expected (theoretical) focal plane coordinates, $(\hat{u}_k, \hat{v}_k)$ ②③, and the **A** matrix ④, (which is denoted by $\frac{\partial(u, v)}{\partial(T_0, L, \phi_0, \lambda_0, h_0, \alpha_0)}$ in Figure 13) and its pseudoinverse ⑤. The differences of the focal plane coordinates, $(\delta u_k, \delta v_k)$, (observed minus theoretical) ⑥ are used to generate adjustments to the tactical parameter estimates by computing ⑦:

$$\begin{pmatrix} \delta T_0 \\ \delta L \\ \delta \phi_0 \\ \delta \lambda_0 \\ \delta h_0 \\ \delta \alpha_0 \end{pmatrix} = [\mathbf{A}^T\mathbf{A}]^{-1}\mathbf{A}^T \begin{pmatrix} \delta u_1 \\ \delta v_1 \\ \delta u_2 \\ \delta v_2 \\ \cdot \\ \cdot \\ \delta u_n \\ \delta v_n \end{pmatrix} \tag{85}$$

If all the elements of the left-hand vector (adjustments to the tactical parameters) are not sufficiently small, the current values (j-1) of the tactical parameters are updated ⑧ to the j^{th}.

$$\begin{pmatrix} T_0 \\ L \\ \phi_0 \\ \lambda_0 \\ h_0 \\ \alpha_0 \end{pmatrix}_j = \begin{pmatrix} T_0 \\ L \\ \phi_0 \\ \lambda_0 \\ h_0 \\ \alpha_0 \end{pmatrix}_{j-1} + \begin{pmatrix} \delta T_0 \\ \delta L \\ \delta\phi_0 \\ \delta\lambda_0 \\ \delta h_0 \\ \delta\alpha_0 \end{pmatrix}_{j-1} \tag{86}$$

If the iteration process has converged, the iteration process is terminated and the current values of the tactical parameters are the best estimates that this process can generate. This entire process may be visualized by a flowchart diagram drawn in Figure 13.

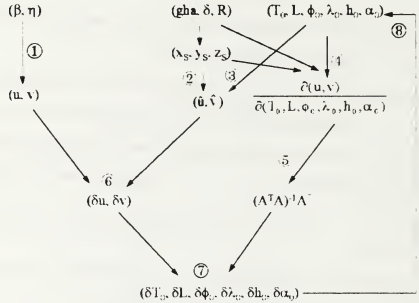


Figure 13. Flowchart Diagram of Iteration Process

G. BURNOUT TIME ESTIMATION

The estimation of burnout time is based on the TBM profile maximum burn time (t_{max}), the last observation time (T_{last}), and the next potential observation time (T_{next}), had it occurred. The maximum burn time according to the profile is:

$$T_{max} = T_0 + t_{max} \tag{87}$$

Two cases can occur:

(a) if $T_{max} \geq T_{next}$ then

$$T_{bo} = T_{last} + \frac{1}{2}(T_{next} - T_{last}) \tag{88}$$

(b) if $T_{max} < T_{next}$ then

$$T_{bo} = T_{last} + \frac{1}{2}(T_{max} - T_{last}) \tag{89}$$

A figure and example are given below as a demonstration:

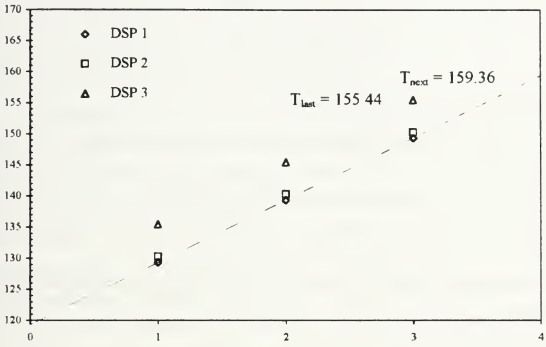


Figure 14. Burnout Time Estimation

For the sample data given in Table 1, using the profile given in Table 2, t_{max} = 62.5 seconds, T_{last} = 155.44, and the next potential observation is T_{next} = 159.36. With T_0 = 109.36, the maximum burn time is T_{max} = 171.86, so case (a) applies and burnout time is estimated at T_{bo} = 157.40.

H. STATE VECTOR GENERATION

The state vector completely defines the TBM's position and velocity, and has six elements. With the tactical parameters and burnout time estimates, generating the state vector at burnout is done by evaluating the position and velocity equations at the burnout time:

$$t_{bo} = T_{bo} - T_0 \tag{90}$$

Substitute t_{bo} for t in (1) and (2), and use (3), (4), and (22) through (25) to get ϕ_{bo}, λ_{bo}, and alt_{bo}:

$$d_{P_{bo}} = a_0 + a_1 t_{bo} + a_2 t_{bo}^2 + a_3 t_{bo}^3 + a_4 t_{bo}^4 \tag{1}$$

$$h_{P_{bo}} = b_0 + b_1 t_{bo} + b_2 t_{bo}^2 + b_3 t_{bo}^3 + b_4 t_{bo}^4 \tag{2}$$

$$d_{bo} = (1 - 1.5L) d_{P_{bo}} \tag{3}$$

$$h_{bo} = (1 + L) h_{P_{bo}} \tag{4}$$

$$\theta_{bo} = \frac{d_{bo}}{r_{eff}} \tag{22}$$

$$\phi_{bo} = \frac{\pi}{2} - \cos^{-1}\left[\cos(\theta_{bo})\sin(\phi_0) + \sin(\theta_{bo})\cos(\phi_0)\cos(\alpha_0)\right] \tag{23}$$

$$\lambda_{bo} = \lambda_0 + \sin^{-1}\left[\frac{\sin(\theta_{bo})\sin(\alpha_0)}{\cos(\phi_{bo})}\right] \tag{24}$$

$$alt_{bo} = h_0 - h_{bo} \tag{25}$$

The burnout velocity is expressed in terms of speed (V_{bo}), flight path angle (γ_{bo}), and heading (α_{bo}):

$$V_{bo} = \sqrt{\dot{d}^2 + \dot{h}^2} \tag{91}$$

$$\gamma_{bo} = \tan^{-1}\left(\frac{\dot{h}}{\dot{d}}\right) \tag{92}$$

$$\alpha_{bo} = \sin^{-1}\left(\frac{\cos(\alpha_0)\sin(\alpha_0)}{\cos(\phi_{bo})}\right) \tag{93}$$

where $\dot{d}$ and $\dot{h}$ are the time derivatives of d and h:

$$\dot{d} = \frac{\partial d}{\partial t} \tag{94}$$

$$\dot{h} = \frac{\partial h}{\partial t} \tag{95}$$

The state vector at burnout, T_{bo}, is:

$$\begin{pmatrix} \phi_{bo} \\ \lambda_{bo} \\ h_{bo} \\ V_{bo} \\ \gamma_{bo} \\ \alpha_{bo} \end{pmatrix} \tag{96}$$

I. CALCULATION OF IMPACT POSITION AND TIME

{*The equations and methods in this section are adapted directly from Chapter 6 of Bate, et al.*} [Ref 6]. The ballistic trajectory is modeled in three phases: powered flight, which is the portion that DSP observes, free-flight, which is a portion of an elliptical orbit, and re-entry, which is the portion from where atmospheric drag becomes significant until missile impact. The powered-flight phase is modeled with the TBM profile polynomials presented earlier. The ellipse traced during free-flight is simulated using inertial two-body mechanics. Atmospheric drag effects during the re-entry phase are not specifically calculated in this model, but are somewhat accounted for by assuming that the distance traveled over the earth from re-entry to impact is the same as from launch to burnout. This is the same as assuming that the earth-central angles are equal:

$$\theta_{re} = \theta_{bo} \tag{97}$$

The speed of the TBM during re-entry is assumed to be unaffected, however. These approximations are recognized artificialities, but are simpler than calculating ballistic coefficients of various TBM's and modeling atmospheric density, and more accurate than assuming the missile remains on its elliptical path until impact. Figure 15 illustrates the geometry and some of the quantities used in the equations.

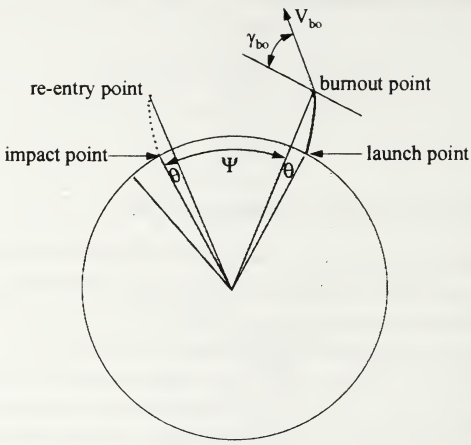


Figure 15. Ballistic Trajectory [Ref. 6]

The equations contained in Chapter 6 of Ref. 6 concern ballistic trajectories and are based upon inertial quantities and a spherical earth model, so the geocentric position vector and inertial speed at burnout are required. Geocentric latitude is generated from (26):

$$\phi'_{bo} = \tan^{-1}[(1-f)^2 \tan(\phi_{bo})] \tag{26}$$

The earth's radius at burnout is calculated with (27):

$$r_{local_{bo}} = \frac{r_e(1-f)}{\sqrt{(1-f)^2 \cos^2(\phi'_{bo}) + \sin^2(\phi'_{bo})}} \tag{27}$$

These values are then transformed into earth-centered, Cartesian coordinates of the TBM:

$$x_{T_{bo}} = \left[r_{local_{bo}} \cos(\phi'_{bo}) + alt_{bo} \cos(\phi_{bo})\right]\cos(\lambda_{bo}) \tag{28}$$

$$y_{T_{bo}} = \left[r_{local_{bo}} \cos(\phi'_{bo}) + alt_{bo} \cos(\phi_{bo})\right]\sin(\lambda_{bo}) \tag{29}$$

$$z_{T_{bo}} = r_{local_{bo}} \sin(\phi'_{bo}) + alt_{bo} \sin(\phi_{bo}) \tag{30}$$

which define the geocentric TBM position at burnout:

$$\mathbf{r}_{bo} = \begin{bmatrix} x_{T_{bo}} \\ y_{T_{bo}} \\ z_{T_{bo}} \end{bmatrix} \tag{8}$$

In an inertial reference frame, the TBM has an initial eastward velocity at launch due to the earth's rotation, v_o. This velocity can be expressed as:

$$v_o = r_{local_{bo}} \omega_e \cos(\phi'_{bo}) \tag{98}$$

where $\omega_e = 15°/\text{hour}$, the rotation rate of the earth. The burnout velocity can be broken down into three inertial (UEN) components. In the inertial frame, the initial eastward velocity is added to the eastward component:

$$\overline{V}_U = V_{bo} \sin(\gamma_{bo}) \tag{99}$$

$$\overline{V}_E = V_{bo} \cos(\gamma_{bo}) \sin(\alpha_{bo}) + v_o \tag{100}$$

$$\overline{V}_N = V_{bo} \cos(\gamma_{bo}) \cos(\alpha_{bo}) \tag{101}$$

Inertial quantities are denoted with a bar, " $\bar{}$ ". The magnitude of this vector is the inertial speed, $\overline{V}_{bo}$:

$$\overline{V}_{bo} = \sqrt{\overline{V}_U^{\,2} - \overline{V}_E^{\,2} + \overline{V}_N^{\,2}} \tag{102}$$

The inertial flight path angle and heading may now be calculated:

$$\overline{\gamma}_{bo} = \sin^{-1}\left(\frac{\overline{V}_U}{\overline{V}_{bo}}\right) \tag{103}$$

$$\overline{\alpha}_{bo} = \tan^{-1}\left(\frac{\overline{V}_E}{\overline{V}_N}\right) \tag{104}$$

Now that the inertial quantities are known, a non-dimensional parameter, Q, is defined as the squared ratio of the inertial speed of the TBM to circular orbit speed at that position:

$$Q = \frac{\overline{V}_{bo}^{\,2} r_{bo}}{\mu} \tag{105}$$

where $\mu = 398601.2 \frac{km^3}{sec^2}$, the gravitational parameter for the earth, and $r_{bo} = |\mathbf{r}_{bo}|$, the magnitude of the TBM position vector.

The eccentricity of the ballistic "orbit" is:

$$e = \sqrt{1 + Q(Q-2)\cos^2(\bar{\gamma}_{bo})} \tag{106}$$

The free-flight "earth central angle", Ψ, is defined by Bate, et al., as that earth-central angle that the missile traverses between burnout and re-entry (see Figure 15). This definition is modified by adding the angle traversed during reentry, θ_{re} to the BMW defined angle, Ψ. This implements the assumption that the TBM's trajectory is modified by drag. The new definition for the angle, Ψ, is:

$$\Psi = 2\cos^{-1}\left(\frac{1 - Q\cos^2(\bar{\gamma}_{bo})}{e}\right) + \theta_{bo} \tag{107}$$

The eccentric anomaly, E, and the semi-major axis, a, of the ballistic trajectory are:

$$E = \cos^{-1}\left(\frac{e - \cos\left(\frac{\Psi}{2}\right)}{1 - e\cos\left(\frac{\Psi}{2}\right)}\right) \tag{108}$$

$$a = \frac{r_{bo}}{2 - Q} \tag{109}$$

Assuming the speed of the TBM is unaffected by drag during re-entry, the time of free flight (burnout to impact) is defined as:

$$t_{ff} = 2\sqrt{\frac{a^3}{\mu}}[\pi - E + e\sin(E)] \tag{110}$$

the time of impact from launch is:

$$t_{im} = t_{bo} + t_{ff} \tag{111}$$

and the time of day of impact is:

$$T_{im} = T_0 + t_{im} \tag{112}$$

Using the Law of Cosines from spherical trigonometry, latitude at impact, ϕ'_{im}, is:

$$\phi'_{im} = \sin^{-1}[\sin(\phi'_{bo})\cos(\Psi) + \cos(2\pi - \bar{\alpha}_{bo})] \tag{113}$$

This can be transformed back into the WGS-84 latitude:

$$\phi_{im} = \tan^{-1}\left(\frac{\tan(\phi'_{im})}{(1-f)^2}\right) \tag{114}$$

This method is not an exact transformation, but it is a close approximation that does not affect the error analysis. A better model should be used if more precise latitude of impact is desired.

Using the Law of Cosines again, and after some algebra, the longitude traversed, $\Delta\lambda$, is:

$$\Delta\lambda = \cos^{-1}\left(\frac{\cos(\Psi) - \sin(\phi'_{im})\sin(\phi'_{bo})}{\cos(\phi'_{im})\cos(\phi'_{bo})}\right) - \omega_e t_{ff} \tag{115}$$

So, the impact longitude, λ_{im}, is:

$$\lambda_{im} = \lambda_{bo} + \Delta\lambda \tag{116}$$

Better models for the ballistic trajectory could possibly be employed. The goal of this analysis is not to precisely determine the actual impact zone, but to determine the effects of error sources upon the result. Thus, the effects found using this model should be applicable to other ballistic trajectory models.

IV. ERROR SOURCES

Noise (error) is present in every aspect of the algorithm presented in Chapter 3. Every measurement, constant, coefficient, model, equation, approximation, and assumption impart some finite quantity of uncertainty upon the result. Even if a quantity is exact to its limit of accuracy, the simple fact that it is represented by a finite number of digits limits the precision of that quantity. For example, if a ruler has one-sixteenth inch gradations, the accuracy of any measurements made with the ruler is ±one-thirty-second of an inch. This means that there are no "perfect" values in the algorithm, and each value used is a source of uncertainty, termed an error source.

Each error source usually has one or more underlying causes. A complete error analysis would break each error source down to its fundamental level, and model each level correctly. An analogy for this is the layers in an onion. The onion can be viewed externally as a whole, but can also be peeled, layer by layer, to reveal deeper, underlying matter that support the external skin. In this section, some of the underlying causes of the overall error sources are identified, but in these analyses, only the overall error magnitudes will be modeled and analyzed.

The obvious origin of any errors present is the observational data since it is the starting point for the algorithm. These data are physical measurements of three types: time, focal plane measurements, and satellite position and orientation.

Time errors can be caused simply by having more than one clock being referenced as a source of time measurement. Imperfect synchronization between clocks' time and time passage rate are obvious error sources. Time delays caused by radio transmission of data due to distance, atmospheric refraction, and relative motion Doppler effects may add another time error. In the extreme, the source of time error can be traced all the way back to our measurement of time passage with respect to the celestial sphere, which is not fixed. The magnitude of time errors is relatively small, and getting smaller as time measurement improvements are being made continually.

The largest source of time error enters the algorithm at the burnout time estimation phase. The 10-second sampling rate limits the accuracy of the estimate to ± 5 seconds for single satellite observations. The accuracy of the profile's value for t_{max} also comes into play in the estimate. The burnout time is crucial to determining the ballistic trajectory, since the TBM is undergoing its greatest acceleration changes during the final seconds of powered flight. Small errors in the estimate of burnout time thus result in larger errors in the estimate of impact time and position.

LOS measurement errors can arise from many noise sources, mainly of two types: attitude uncertainties and IR radiation measurement errors. The attitude of the spacecraft needs to be precisely known (and it simplifies the process if it is very stable). A 28 microradian error in pitch or roll pointing accuracy of a geosynchronous satellite equates to a 1 kilometer error on the surface of the earth, at least 35786 kilometers away. Any attitude control system inaccuracy effects are amplified by the geosynchronous altitude. Vibrational noise from spinning momentum wheels, uncertain knowledge of the mass properties of the yaw-spinning satellite, star catalog and star sensor measurement errors, thruster misalignments and disturbance torques, tachometer errors, and mechanical misalignments all contribute to the total attitude uncertainty. The knowledge of the telescope boresight axis alignment with the satellite's reference frame is defined by the design and manufacturing of the DSP satellite, and changes slightly with thermal variations.

The IR radiation measurements of intensity and angle of arrival (AOA) also have several underlying error sources. Locations of the individual photoelectric cells on the focal plane array are recorded in what is termed the "Focal Plane Vector Table" (FPVT). The positions of the PEC's change as the satellite heats and cools with varying sun-satellite orientations, causing a warping of the focal plane, but the FPVT does not account for the changes real-time. In addition, detector noise, refraction, and IR attenuation due to clouds and water vapor all add to the total measurement error.

Satellite position measurements from AFSCN are used to update orbit ephemeris data. The measurements are, of course, inexact, but those errors are compounded over time because the updates occur only once weekly. The ephemeris data is propagated to

determine satellite position during the week between updates. DSP is a geosynchronous satellite, so its orbit does not change greatly in that period, but errors in the propagation model can cause the predicted position to be different from reality.

Keeping all of these underlying components in mind, the errors are modeled in the tactical parameter estimation algorithm. The real-world error component magnitudes, bias and random distributions may be different from those simulated, but the overall effects manifest themselves in a manner close to the error models. It should be possible to extrapolate the results obtained in this study to different errors by properly scaling the different magnitudes and distributions. It must be emphasized that since the goal of this study is not to exactly determine the tactical parameters at launch, state vector at burnout, or impact time and position. The purpose is to determine the contribution of the error sources upon these values, and this can be done with approximate models for the errors. If the intended goal is to calibrate the system to eliminate bias errors, then the sources of the errors must be determined more exactly to determine what is observable (measurable).

V. NUMERICAL ANALYSIS

The algorithm described in Chapter III was programmed into MATLAB™. Three error sources (time, satellite position, and LOS) were simulated and added to the observational data, first separately, and then combined. The effects of the errors upon the results are analyzed at three points: launch position, burnout position, and impact position. Five Middle Eastern capital cities were chosen as fictitious launch sites to determine the effects of TBM launch position upon the accuracy of the results.

The model for time error is a random uniform distribution between 0 and 1 milliseconds. For satellite position, a random normal distribution with zero mean and standard deviation of 200 meters was added to each of the components (R, gha, δ). This is approximately a one standard deviation sphere with radius 346 meters. The LOS error was modeled as a random normal distribution with a standard deviation of 5 microradians, added to both β and η components. Histograms of sample error distributions are shown in the following three figures.

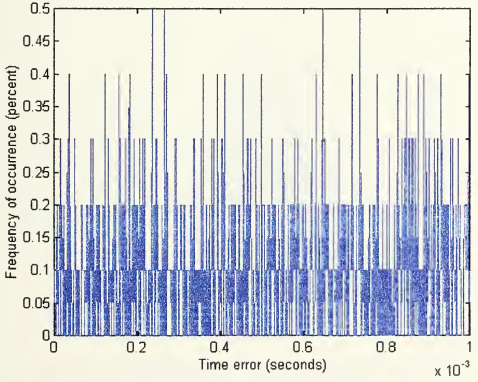


Figure 16. Histogram of Time Error - Uniform Distribution

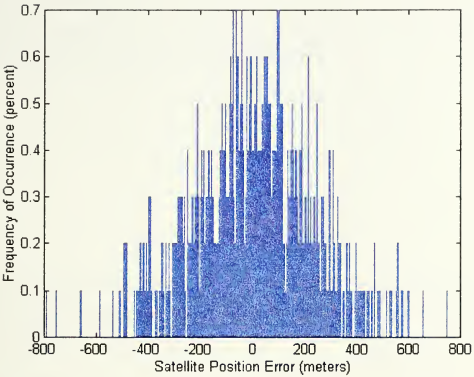


Figure 17. Histogram of Satellite Position Error - Normal Distribution

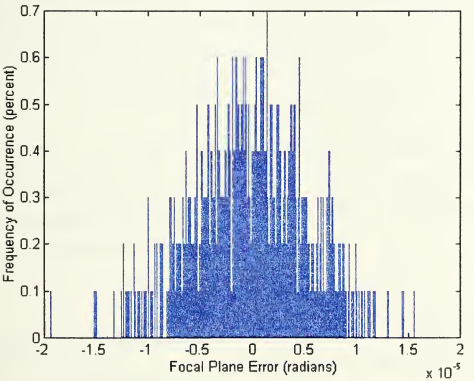


Figure 18. Histogram of LOS Error - Normal Distribution

Errors added to the observations had the effect of creating an ellipse of incorrect data points around the true position, termed an "error ellipse". The 1-σ ellipses shown in the next figures are determined using statistical methods.[Ref. 7] For the launch ellipse, a matrix of two columns was collected during the simulation:

$$\text{lpts} = \begin{bmatrix} \lambda_{0_{true}} & \phi_{0_{true}} \\ \lambda_{0_{error=0}} & \phi_{0_{error=0}} \\ \lambda_{0_{error\,1}} & \phi_{0_{error\,1}} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ \lambda_{0_{error\,999}} & \phi_{0_{error\,999}} \end{bmatrix} \tag{117}$$

where lpts = launch points, the matrix of launch longitudes, λ_0, and latitude, ϕ_0. The first row is the generated true launch position, the second row is the calculated launch position with no error added to the observational data, and the remaining 999 rows are the calculated launch positions with errors added to the observational data. The first two rows are error-free, and are removed to produce a matrix of only "corrupt" positions. To determine the ellipse dimensions, a and b (semi-major and semi-minor axes lengths), the eigenvectors and values of the covariance of this matrix were calculated. Twice the square root of the eigenvalue diagonal elements produces a and b:

$$\begin{pmatrix} a \\ b \end{pmatrix} = 2 * \begin{bmatrix} \sqrt{\text{eigenvalue}(1,1)} & 0 \\ 0 & \sqrt{\text{eigenvalue}(2,2)} \end{bmatrix} \tag{118}$$

and the eigenvector is the direction cosine matrix for rotating the ellipses from the primary axes. The impact ellipses are calculated similarly, with the initial data matrix being composed of impact longitudes and latitudes, instead of launch. The ellipses can then be plotted using the ellipse equation:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{119}$$

and letting x vary between ±a, and letting y be the dependent variable. Multiplying the resultant (x, y) coordinates by the eigenvector matrix rotates them to the correct orientations. The semi-axes are in units of radians, since the positional values are in

radians. Thus, the (x, y) plots produced are the longitude and latitude of the points, once the values are converted to degrees.

For the burnout position, the same method is used, but the data point matrix has three columns, the third being the altitude of the TBM at burnout. The covariance method works as well in three dimensions, producing a, b, and c, the three semi-axes for the ellipsoid formed. The coordinates are calculated in the same manner, using the equation for an ellipsoid:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1 \tag{120}$$

and, again, rotating by pre-multiplying by the eigenvector matrix.

The following figures are representative of all the simulated cases, but this particular case is a 300-km TBM launched from Baghdad heading 0°, with the combined erorrors.

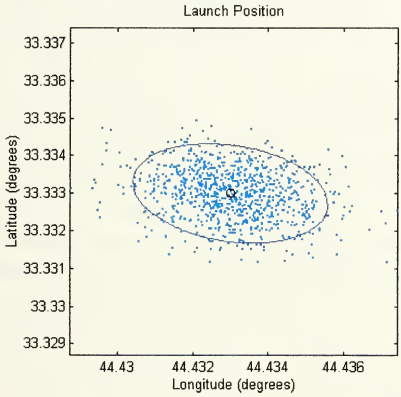


Figure 19. Launch Position Error Ellipse

The ellipse drawn on top of the data points encompasses the data points within one standard deviation of the mean, assuming the distribution is normal. Since the time error

has a uniform distribution, 1-σ is not the expected 66% for time error cases. This method is used anyway for lack of a better one and to maintain a common baseline for comparison. As it turns out, the time error contribution is very small, and thus does not effect the combined error case distribution.

The position at burnout is plotted in three dimensions, using altitude as the third coordinate to form an ellipsoidal error volume.

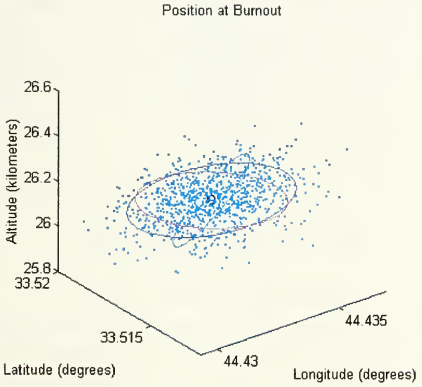


Figure 20. Position at Burnout

The three ellipses show the outline of the ellipsoidal volume. It is difficult to perceive the ellipse orientations, but they are mutually orthogonal, (necessarily, since they correspond to the eigenvectors of distinct eigenvalues).

The impact ellipse is similar but larger than the launch position ellipse, due to the propagation of velocity errors from burnout to impact and the error in the burnout time.

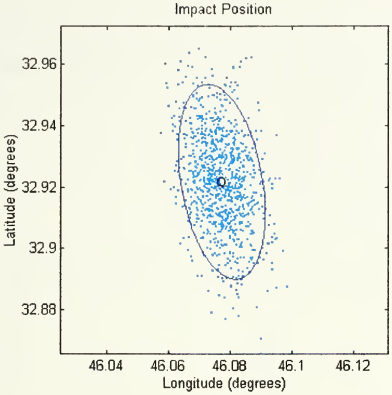


Figure 21. Impact Position Error Ellipse

The three previous figures illustrate the error ellipses(oids), but showing all the data obtained in this format is unwieldy. The sizes (areas and volumes) of the positional errors are the important quantities for identifying which error source has the largest effect. Thus, the data have been processed to determine these quantities, and are plotted next.

The data for the 300-km TBM was collated by city, heading, and error source. The ellipse(oid) dimensions calculated must be equated to the actual distance on the ground to determine the area of the ellipses. To do this, the angles must be in radians, and are multiplied by r_{local}. The distance between lines of longitude shrinks as the latitude moves away from the equator, so the longitudinal distance must be multiplied by the cosine of the geocentric latitude, ϕ'. Thus the equation for ellipse area becomes:

$$\text{area} = \pi r_{local}^{\,2}\, ab \cos(\phi') \tag{121}$$

The ellipsoid volume is not as simple, so a parallelepiped with dimensions, a x b x c, is used to approximate it:

$$\text{volume} = abc \tag{122}$$

These areas and volumes are plotted in the following figures, separated by error source and heading. Each launch point is represented by a different color:

Aden = blue

Baghdad = black

Damascus = green

Riyadh = red

Tehran = cyan

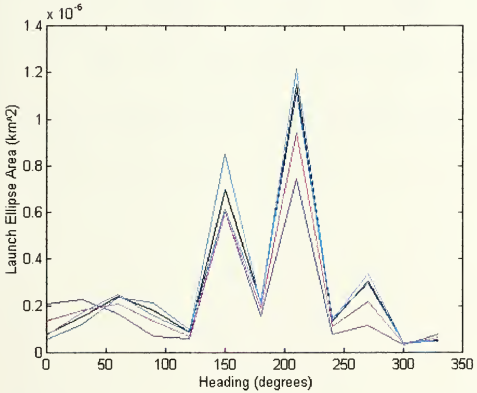


Figure 22. Time Error Effects Upon Launch Ellipse Area

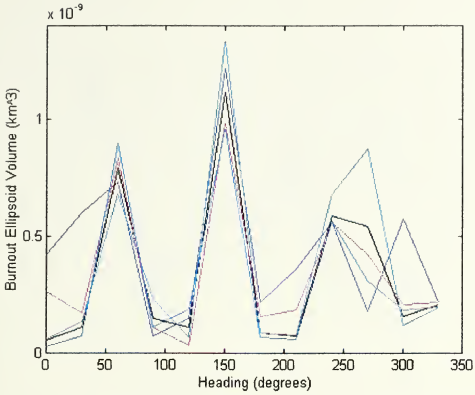


Figure 23 Time Error Effects Upon Burnout Ellipsoid Volume

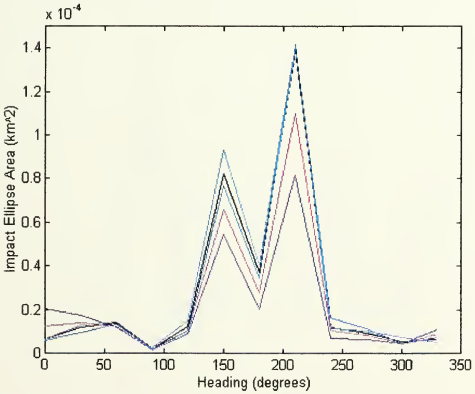


Figure 24. Time Error Effects Upon Impact Ellipse Area

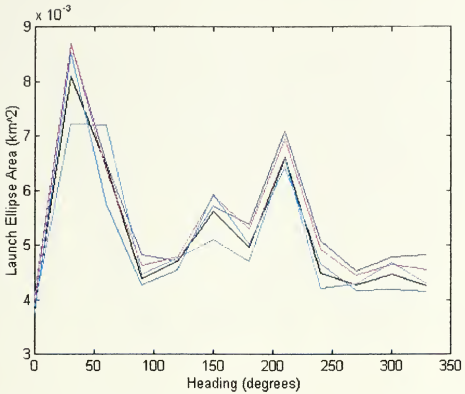


Figure 25. Satellite Position Error Effects Upon Launch Ellipse Area

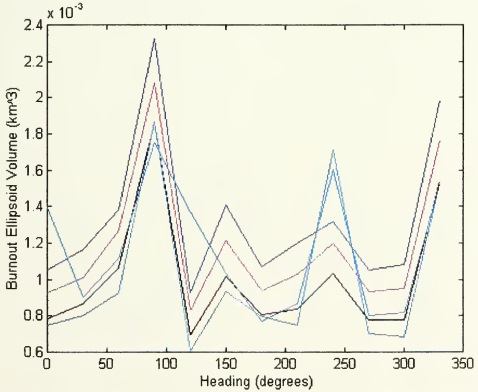


Figure 26. Satellite Position Error Effects Upon Burnout Ellipsoid Volume

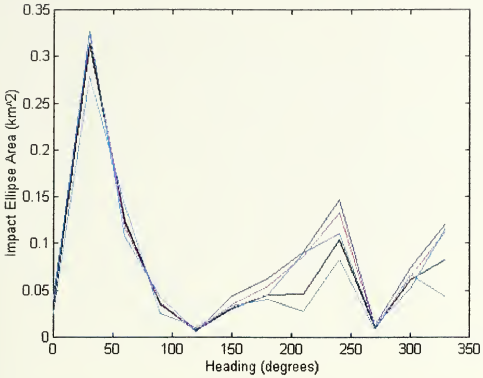


Figure 27. Satellite Position Error Effects Upon Impact Ellipse Area

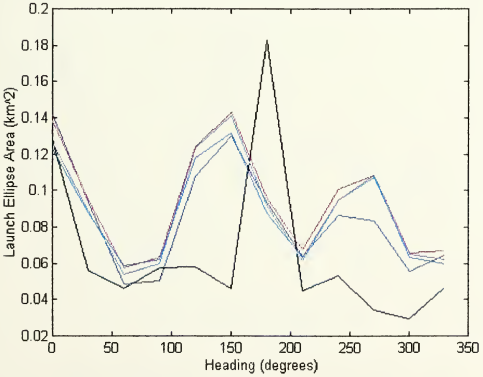


Figure 28. LOS Error Effects Upon Launch Ellipse Area

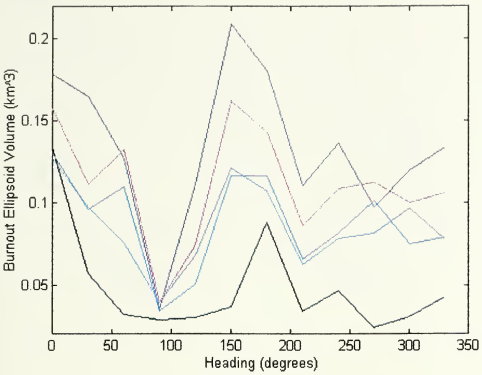


Figure 29. LOS Error Effects Upon Burnout Ellipsoid Volume

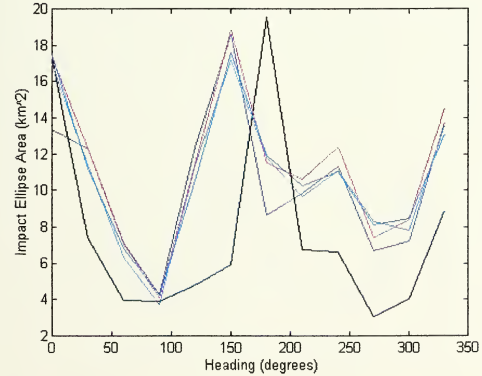


Figure 30. LOS Error Effects Upon Impact Ellipse Area

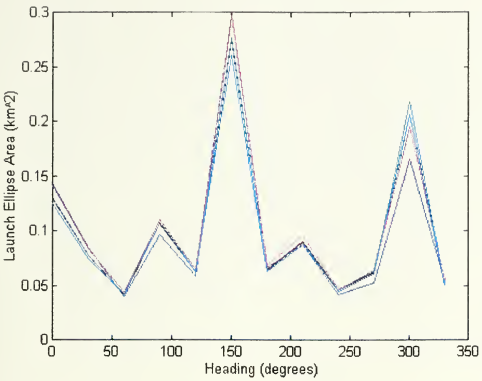


Figure 31 Combined Error Effects Upon Launch Ellipse Area

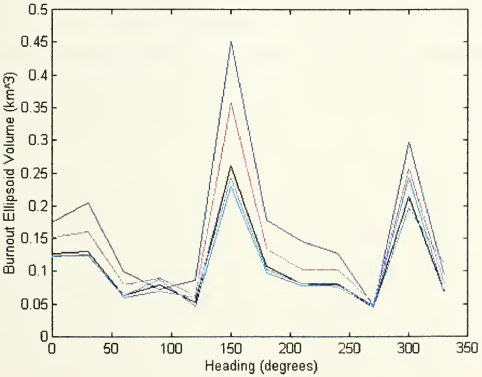


Figure 32. Combined Error Effects Upon Burnout Ellipsoid Volume

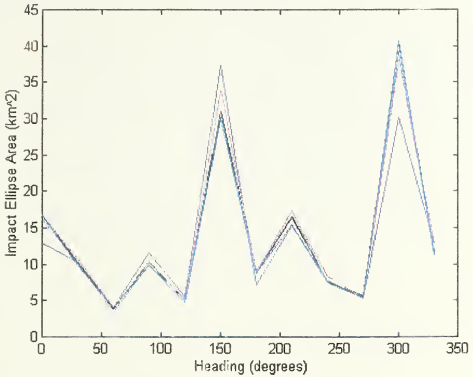


Figure 33. Combined Error Effects Upon Impact Error Ellipse

It is immediately apparent from these plots that the launch site has little effect upon the size of the error ellipses and ellipsoids. It is also obvious that the size is dependent upon TBM heading and observation geometry. The relative magnitudes of the error effects are not as obvious, so a plot of four error ellipses is shown in the next figure, with each ellipse formed by a different error source.

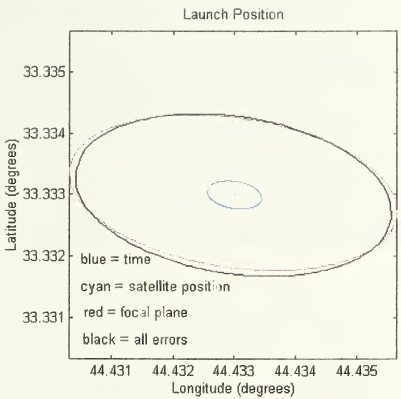


Figure 34. Launch Error Ellipses for Each Error Source

From this plot it is easily recognized that the errors can be ranked in order of effect:

1. focal plane errors
2. satellite position errors
3. time

All the error sources together produce the largest ellipse, as would be expected from the superposition principle.

This is further illustrated by Table 3. Each error source is a row: "time" is the time error runs, "satpos" is the satellite position error runs, "LOS" is line of sight, and "combin." is the combined errors runs.

ERROR	LAUNCH ELLIPSE AREA (km^2)			BURNOUT ELLIPSOID VOLUME (km^3)			IMPACT ELLIPSE AREA (km^2)		
	min	mean	max	min	mean	max	min	mean	max
time	3.26e-8	2.59e-7	1.22e-6	2.9e-11	3.6e-10	1.33e-9	1.57e-6	2.61e-5	1.41e-4
satpos	3.77e-3	5.27e-3	8.68e-3	6.04e-4	1.14e-3	2.32e-3	6.12e-3	7.92e-2	3.27e-1
LOS	2.97e-2	8.45e-2	1.83e-1	2.37e-2	9.33e-2	2.09e-1	3.05e0	1.02e1	1.95e1
combin.	3.91e-2	1.01e-1	2.97e-1	4.44e-2	1.25e-1	4.50e-1	3.35e0	1.38e1	4.06e1

Table 3. Error Ellipse Areas and Ellipsoid Volumes

This is also shown graphically in the following plots:

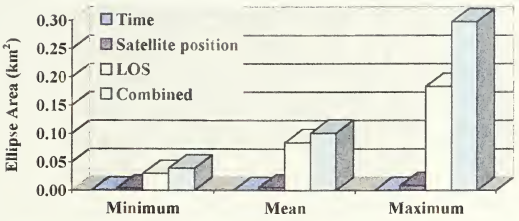


Figure 35. Launch Ellipse Area Comparison

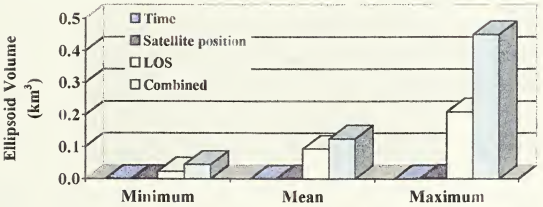


Figure 36. Burnout Ellipsoid Volume Comparison

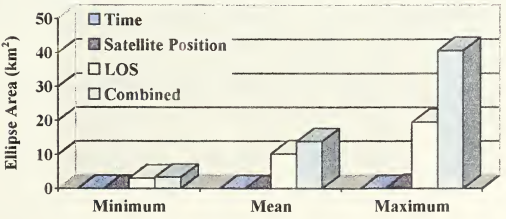


Figure 37. Impact Ellipse Area Comparison

The sum of the error sources' positional error sizes is less than the combined error source's positional error size. This implies a nonlinear interaction of error effects with each other. This is a reasonable result considering the complexity of the simulated system, and noting that most real systems exhibit nonlinear behavior.

Taking advantage of the ease of modifying MATLAB™ code, various changes to the present DSP system model were put into simulation to determine the effects upon the accuracy of the results. For every case, Baghdad is the launch site and the observational data were modified by combined error sources.

The first case is the control case with nominal system parameters. It is listed as "Bagcom" to represent "Baghdad combined errors", and is used as a baseline case for comparison. The second case, "Synchr." shows the effects of synchronizing the spins of the satellites so that the satellites scan a single area of interest within one second of each other. For the third case, "Molniya", the satellite at 70° GHA was elevated to 63.4° declination, to simulate a Molniya + geostationary viewing geometry. The remaining cases were simulating faster scan rates from 10 seconds down to 2.5 seconds. The numerical results are shown in Table 4.

ERROR	LAUNCH ELLIPSE AREA (km^2)			BURNOUT ELLIPSOID VOLUME (km^3)			IMPACT ELLIPSE AREA (km^2)		
	min	mean	max	min	max	max	min	mean	max
Bagcom	4.21e-2	1.01e-1	2.74e-1	4.46e-2	1.09e-1	2.60e-1	3.83e0	1.40e1	4.04e1
Synchr.	4.40e-2	7.07e-2	1.10e-1	2.09e-2	4.89e-2	8.61e-2	3.94e0	7.82e0	1.52e0
Molniya	4.39e-2	7.58e-2	1.25e-1	2.61e-2	5.57e-2	1.04e-1	2.91e0	7.96e0	1.85e1
10 s sc	4.21e-2	1.01e-1	2.74e-1	4.46e-2	1.09e-1	2.60e-1	3.83e0	1.40e1	4.04e1
7.5 s sc	2.62e-2	7.44e-2	1.95e-1	2.53e-2	4.55e-2	1.09e-1	4.03e0	8.72e0	2.08e1
5 s sc	2.48e-2	5.02e-2	1.52e-1	1.23e-2	2.52e-2	6.52e-2	2.52e0	5.84e0	1.64e1
2.5 s sc	1.18e-2	2.36e-2	3.48e-2	5.05e-3	9.21e-3	1.31e-2	1.03e0	2.95e0	4.58e0

Table 4. Various Case Comparison

The effects of the modifications are more easily observed graphically:

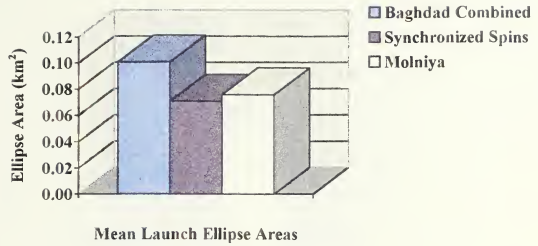


Figure 38. Comparison of Mean Launch Ellipse Areas

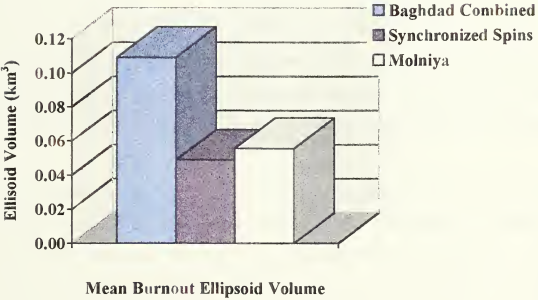


Figure 39. Comparison of Mean Burnout Ellipsoid Volumes

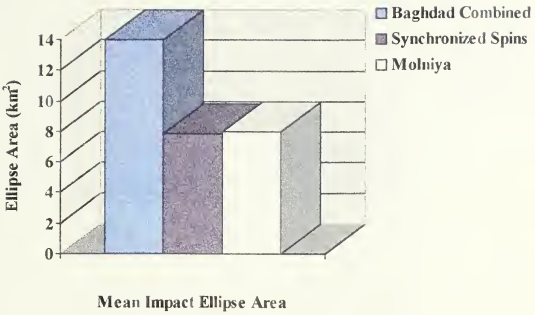


Figure 40. Comparison of Mean Impact Ellipse Areas

Both modifications have the effect of decreasing the areas and volume by about a third. This is the expected result for the Molniya case, since the viewing geometry is more three-dimensional with two satellites on the equator and the middle satellite near Molniyan apogee.

The synchronized spin results were a surprise, however. The effect upon the launch ellipse area is fine, but since the burnout time estimation should have been less accurate, the burnout ellipsoid volume and impact ellipse area were expected to remain unchanged at best. Obtaining simultaneous observations would have obvious advantages, since they could be processed differently and be used to determine the position exactly for discrete instants in time during the boost trajectory. However, these observations were not processed differently, and were not simultaneous -- only close together (within one second of each other). Still, the result may be a manifestation of some mathematical process that causes the increase in accuracy.

The following plots show the effects of increasing the scan rate, or alternately, decreasing the time between observations. This allows more observations to be made during the boost phase.

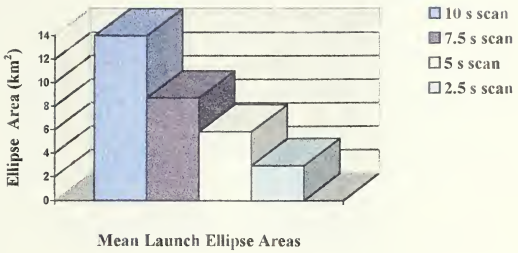


Figure 41. Scan Rate Effects on Launch Ellipse Area

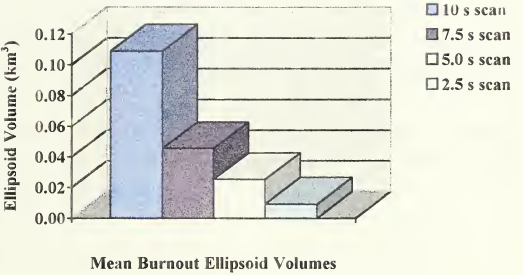


Figure 42. Scan Rate Effects on Burnout Ellipsoid Volume

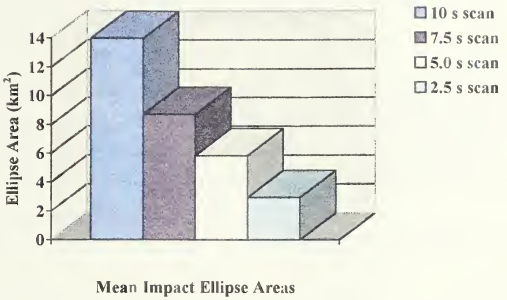


Figure 43. Scan Rate Effects on Impact Ellipse Area

The effects seem to be linear decreases upon the launch and impact ellipse areas, and quadratic decreases on the burnout ellipsoids. This result is in perfect agreement with intuitive predictions.

In summary, the numerical results are generally in accordance with expected results. As a qualitative check of the quantitative results, the next chapter is devoted to a qualitative analysis of a simplified case for comparison.

VI. QUALITATIVE ANALYSIS

Starting with the equations presented in Chapter 3, a qualitative analysis has been done to determine the expected trends in the numerical analysis. The problem is simplified by assuming that the equations are exact and that there is only one satellite. The effects of time errors are not analyzed, so the problem is to determine how satellite position and focal plane error effects should effect the results.

TBM geocentric longitude and latitude can be expressed in (XYZ) coordinates:

$$\lambda = \tan^{-1}\left(\frac{y}{x}\right) \tag{11}$$

$$\phi' = \tan^{-1}\left(\frac{z}{\sqrt{x^2 + y^2}}\right) \tag{12}$$

These (XYZ) coordinates can be expressed in terms of satellite position and focal plane coordinates using the LOS projection equation (8):

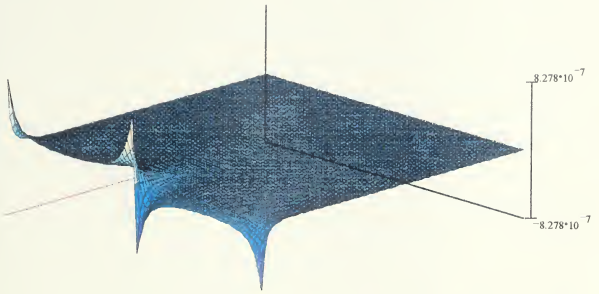
$$\mathbf{r} = \begin{pmatrix} x_I \\ y_T \\ z_I \end{pmatrix} = \mathbf{R}_s + \rho\hat{\mathbf{e}} \tag{8}$$

Rewriting the terms on the left-hand side in terms of R, gha, δ, η, and β generates a rather lengthy equation, and is given in Appendix B. The next task is to find the partial derivatives of the TBM position with respect to the five error terms, using the Chain Rule. Taking the partial with respect to R, for example:

$$\frac{\partial\lambda}{\partial R} = \frac{\partial\lambda}{\partial x}\frac{\partial x}{\partial R} + \frac{\partial\lambda}{\partial y}\frac{\partial y}{\partial R} + \frac{\partial\lambda}{\partial z}\frac{\partial z}{\partial R} \tag{123}$$

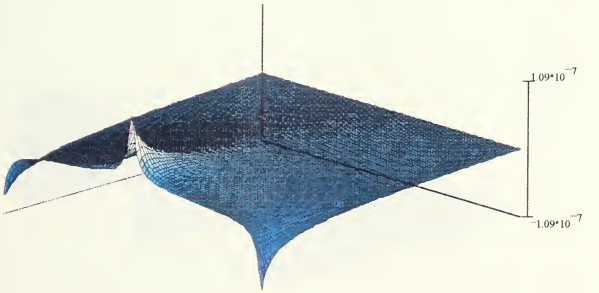
$$\frac{\partial\phi'}{\partial R} = \frac{\partial\phi'}{\partial x}\frac{\partial x}{\partial R} + \frac{\partial\phi'}{\partial y}\frac{\partial y}{\partial R} + \frac{\partial\phi'}{\partial z}\frac{\partial z}{\partial R} \tag{124}$$

Again, expression this equation written explicitly in terms of the error sources becomes tedious, and can be found in Appendix B. The results, however, are interesting and are shown as plots. The horizontal axis pointing left is η, the horizontal axis pointing right is β, and the vertical axis is the differential.



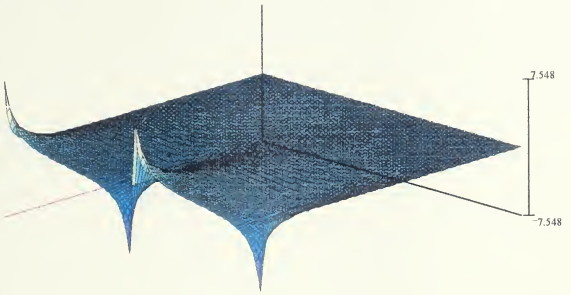
dλ dR

Figure 44. $\frac{\partial \lambda}{\partial R}$



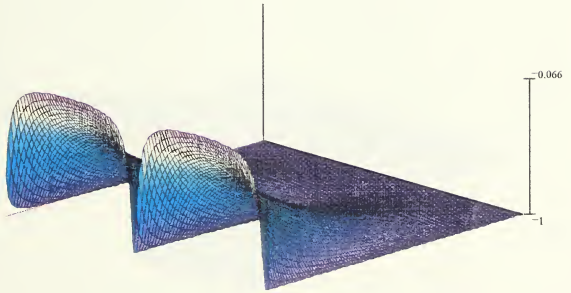
dϕ 'dR

Figure 45. $\frac{\partial \phi'}{\partial R}$



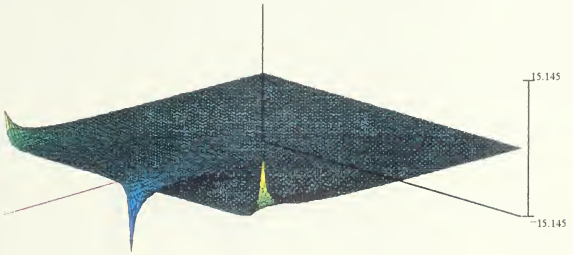
$d\lambda.d\delta$

Figure 46. $\frac{\partial \lambda}{\partial \delta}$



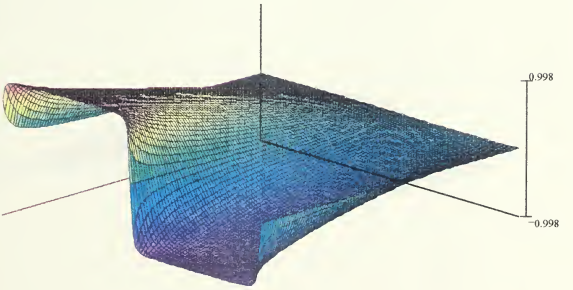
$d\phi'd\delta$

Figure 47. $\frac{\partial \phi'}{\partial \delta}$



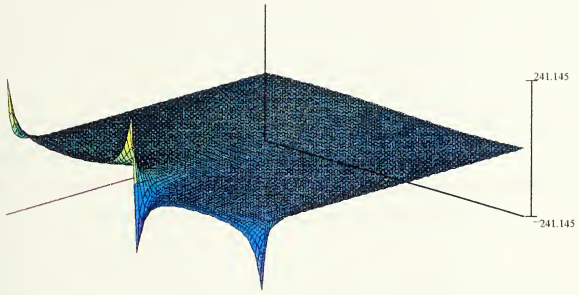
$d\lambda\, d\beta$

Figure 48 $\dfrac{\partial \lambda}{\partial \beta}$



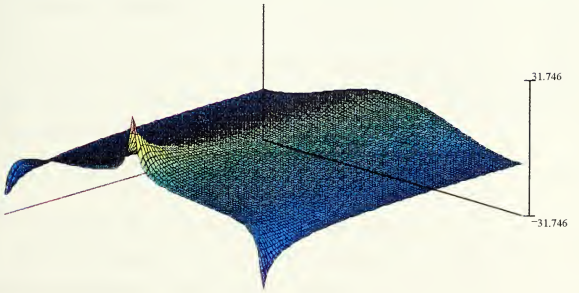
$d\phi'\, d\beta$

Figure 49. $\dfrac{\partial \phi'}{\partial \beta}$



dλ,dη

Figure 50. $\frac{\partial \lambda}{\partial \eta}$



dϕ'dη

Figure 51. $\frac{\partial \phi'}{\partial \eta}$

The partial derivatives with respect to gha turn out to be simple:

$$\frac{\partial \lambda}{\partial \text{gha}} = 1 \tag{125}$$

$$\frac{\partial \phi'}{\partial \text{gha}} = 0 \tag{126}$$

The observations one can make from analyzing these plots is that most of the partial derivative plots are flat and close to zero. When ηapproaches 8.6°, however, some of the values begin to get large. This happens when the observed IR event is close to the limb of the earth as viewed from a geosynchronous satellite. There also appear to be transitions when η approaches 8.6° and b is 0° or 180°.

Figures 44 and 45 show that satellite radius error effects are of such small magnitude that they are neglible. Figures 46 and 47 show that satellite declination error effects are more important to determine latitude than longitude, except when η approaches 8.6°. Figures 48 and 49 show that β LOS errors affect latitude determination more than longitude, except, again, when η approaches 8.6°. Figures 50 and 51 show that η LOS errors have the largest effect of all, and the partials exhibit the same behavior near the limb.

These plots of formulae give an indication of what magnitude the error ellipse should have, given the magnitude of the error is known, by using that magnitude in equations similar to (127).

$$\Delta\lambda = \frac{\partial \lambda}{\partial R}\Delta R = \left(\frac{\partial \lambda}{\partial x}\frac{\partial x}{\partial R} + \frac{\partial \lambda}{\partial y}\frac{\partial y}{\partial R} + \frac{\partial \lambda}{\partial z}\frac{\partial z}{\partial R}\right)\Delta R \tag{127}$$

where R can be any of the five error quantites (R, δ, gha, η, or β).

To test this assumption, substitute the magnitude of the errors used in the simulation, and compare the result with the numerical analysis. The errors are:

$$\Delta R = 200 \text{ m} \tag{128}$$

$$\Delta\delta = \tan^{-1}\left(\frac{0.200 \text{ km}}{42164.17 \text{ km}}\right) \tag{129}$$

$$\Delta gha = \tan^{-1}\left(\frac{0.200\ km}{42164.17\ km}\right) \tag{130}$$

$$\Delta\eta = 5\ \mu radians \tag{131}$$

$$\Delta\beta = 5\ \mu radians \tag{132}$$

The generalized equations for the partial derivatives of the geocentric longitude and latitude can be used to find the derivatives for a specific location by using specific focal plane coordinates. For Baghdad, viewed from the satellite at 70° gha, in particular, these coordinates would be:

$$\eta = 0.1216\ radians \tag{133}$$

$$\beta = 3.8546\ radians \tag{134}$$

Using these values, the partial derivatives with respect to the five error sources become:

$$\frac{\partial\lambda}{\partial\beta} = -0.573\,\frac{rad}{rad} \tag{135}$$

$$\frac{\partial\phi'}{\partial\beta} = -0.753\,\frac{rad}{rad} \tag{136}$$

$$\frac{\partial\lambda}{\partial\eta} = -13.788\,\frac{rad}{rad} \tag{137}$$

$$\frac{\partial\phi'}{\partial\eta} = 5.96\,\frac{rad}{rad} \tag{138}$$

$$\frac{\partial\lambda}{\partial\delta} = 0.656\,\frac{rad}{rad} \tag{139}$$

$$\frac{\partial\phi'}{\partial\delta} = -0.658\,\frac{rad}{rad} \tag{140}$$

$$\frac{\partial\lambda}{\partial R} = -3.663\ x10^{-8}\,\frac{rad}{m} \tag{141}$$

$$\frac{\partial\phi'}{\partial R} = 1.583\ x10^{-8}\,\frac{rad}{m} \tag{142}$$

$$\frac{\partial\lambda}{\partial gha} = 1\,\frac{rad}{rad} \tag{143}$$

$$\frac{\partial \phi'}{\partial \text{gha}} = 0 \frac{\text{rad}}{\text{rad}} \tag{144}$$

Multiplying (135) through (144) by the appropriate error quantities produces the geocentric longitude and latitude error values.

$$\Delta\phi'_R = 3.166 \times 10^{-6} \text{ rad} \tag{145}$$

$$\Delta\phi'_{gha} = 0 \text{ rad} \tag{146}$$

$$\Delta\phi'_\delta = -3.121 \times 10^{-6} \text{ rad} \tag{147}$$

$$\Delta\phi'_\eta = 2.98 \times 10^{-5} \text{ rad} \tag{148}$$

$$\Delta\phi'_\beta = -3.765 \times 10^{-6} \text{ rad} \tag{149}$$

$$\Delta\lambda_R = -7.326 \times 10^{-6} \text{ rad} \tag{150}$$

$$\Delta\lambda_{gha} = 4.743 \times 10^{-6} \text{ rad} \tag{151}$$

$$\Delta\lambda_\delta = -3.112 \times 10^{-6} \text{ rad} \tag{152}$$

$$\Delta\lambda_\eta = -6.894 \times 10^{-5} \text{ rad} \tag{153}$$

$$\Delta\lambda_\beta = -2.865 \times 10^{-6} \text{ rad} \tag{154}$$

Grouping these in terms of satellite position and focal plane and summing the absolute values:

$$\Delta\phi'_{\text{satellite position error}} = \Delta\phi'_R + \Delta\phi'_{gha} + \Delta\phi'_\delta = 6.287 \times 10^{-6} \text{ rad} \tag{155}$$

$$\Delta\lambda_{\text{satellite position error}} = \Delta\lambda_R + \Delta\lambda_{gha} + \Delta\lambda_\delta = 1.518 \times 10^{-5} \text{ rad} \tag{156}$$

$$\Delta\phi'_{\text{focal plane error}} = \Delta\phi'_\eta + \Delta\phi'_\beta = 3.357 \times 10^{-5} \text{ rad} \tag{157}$$

$$\Delta\lambda_{\text{focal plane error}} = \Delta\lambda_\eta + \Delta\lambda_\beta = 7.181 \times 10^{-5} \text{ rad} \tag{158}$$

These values are in radians, which can be equated to kilometers on the surface of the earth by multiplying latitude by the earth's radius and longitude by the earth's radius and the cosine of the latitude. For this analysis, earth's radius is assumed to be 6378.137 kilometers:

$$\Delta\phi'_{\text{satellite position error}} = (6.287 \times 10^{-6} \text{ rad})\, 6378.137 \text{ km} = 4.010 \times 10^{-2} \text{ km} \tag{159}$$

$$\Delta\lambda_{\text{satellite position error}} = (1.518 \times 10^{-5} \text{ rad}) \cos(33.333^\circ) 6378.137 \text{ km} = 8.089 \times 10^{-2} \text{ km} \tag{160}$$

$$\Delta\phi'_{\text{focal plane error}} = (3.357 \times 10^{-5} \text{ rad})\, 6378.137 \text{ km} = 2.141 \times 10^{-1} \text{ km} \quad (161)$$

$$\Delta\lambda_{\text{focal plane error}} = (7.181 \times 10^{-5} \text{ rad}) \cos(33.333^{\circ})\, 6378.137 \text{ km} = 3.827 \times 10^{-1} \text{ km} \quad (162)$$

Assuming that these values are equivalent to the semi-axes of an error ellipse, multiplying longitude error by latitude error and by π gives the area of the error ellipses:

$$\text{ellipse area}_{\text{satellite position error}} = 1.019 \times 10^{-2} \text{ km}^2 \quad (163)$$

$$\text{ellipse area}_{\text{focal plane error}} = 2.574 \times 10^{-1} \text{ km}^2 \quad (164)$$

The focal plane error effects are 25 times greater than the effects of the satellite position error. The relative magnitudes agree with the numerical results, but the specific values obtained here are greater than those obtained numerically by a factor of about three. The numerically obtained values were:

$$\text{launch ellipse mean area}_{\text{satellite position error}} = 5.27 \times 10^{-3} \text{ km}^2 \quad (165)$$

$$\text{launch ellipse mean area}_{\text{focal plane error}} = 8.45 \times 10^{-2} \text{ km}^2 \quad (166)$$

This difference is expected, since this analysis does not account for stereo viewing, the least squares iteration process, or time effects.

This result shows that it is possible to compute error ellipse areas using a simplified qualitative analysis. The relative order of the magnitude of the effects is in agreement with predictions, and verifies the results obtained numerically. To do the same analysis for burnout volumes and impact areas would require the addition of time errors, and trajectory equations. This increases the difficulty of this "back of the envelope" analysis, and is not treated here.

VII. CONCLUSION

The TALON SHIELD / ALERT state vector estimation algorithm was correctly modeled in MATLAB™. System errors were modeled and introduced into the algorithm to determine the effects upon the accuracy of the final results. A simplified qualitative analysis verified the quantitative results.

The three questions posed in the first chapter are now answered:

1. The errors in the system are broadly categorized as errors in time, satellite position, and LOS.
2. The relative magnitudes of the error effects listed largest to least:
   A. LOS - using a zero-mean, 5 μradian standard deviation normal error distribution in focal plane coordinates
   B. satellite position - using a zero-mean, 200 meter standard deviation normal error distribution in satellite positon coordinates
   C. time - using a uniform error distribution between zero and 1 millisecond
3. The effects seem to behave independently, since the principle of superposition works. The effects of the combined errors are slightly greater than the sum of the individual effects.

Additional runs were made to determine the effects of various geometries, spin rates, and observation synchronization.

There are many tasks left to accomplish in this area of research. The MATLAB code can be optimized, additional parameters and system changes can be simulated, and more detailed analysis of the results can be made. The error sources can be modeled more exactly by analytically "peeling the error onion", or by using actual DSP values and measurements. The results obtained could be further analyzed, possibly to

Simulating the DSP system with computer code has proven to be an effective tool for error analysis, which is a necessary first step for determining how best to improve the present TBM detection system or modify the design of a follow-on system.

APPENDIX A. "A" MATRIX DERIVATION

The A matrix is made up of partial derivatives of focal plane coordinates with respect to the tactical parameters. These elements can be determined qualitatively. Exact equations are not required for the iteration process to converge, so small-angle assumptions are made to simplify the resulting equations. The benefit of using these approximations for the matrix elements is the ease of computation and thus a reduction of computing time.

The A matrix is divided into three components, $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$. The elements of $\mathbf{A}_1$, and $\mathbf{A}_2$ will be derived in order, using MATHCAD 5.0 Plus. The $\mathbf{A}_3$ matrix is derived in Chapter 3. The elements of the first column of $\mathbf{A}_1$ are partials of latitude, ϕ:

$$\phi = \frac{\pi}{2} - \operatorname{acos}\left(\cos(\theta)\cdot\sin(\phi_0) + \sin(\theta)\cdot\cos(\phi_0)\cdot\cos(\alpha_0)\right)$$

Taking the partial derivatives:

$$\frac{d\phi}{d\theta} = \frac{\sin(\theta)\cdot\sin(\phi_0) - \cos(\theta)\cdot\cos(\phi_0)\cdot\cos(\alpha_0)}{\sqrt{1 - \left(\cos(\theta)\cdot\sin(\phi_0) - \sin(\theta)\cdot\cos(\phi_0)\cdot\cos(\alpha_0)\right)^2}}$$

Using the small angle assumption that $\sin(\theta)\cong 0$ and $\cos(\theta)\cong 1$:

$$\frac{d\phi}{d\theta} = \frac{\cos(\phi_0)\cdot\cos(\alpha_0)}{\sqrt{1 - \sin(\phi_0)^2}} = \cos(\alpha_0)$$

$$\theta = \frac{d}{r_{eff}}$$

$$\frac{d\theta}{dd} = \frac{1}{r_{eff}}$$

$$d = (1 - 1.5\cdot L)\cdot d_p$$

$$d_P = a_0 + a_1\cdot(T - T_0) + a_2\cdot(T - T_0)^2 + a_3\cdot(T - T_0)^3 + a_4\cdot(T - T_0)^4$$

$$\frac{dd}{dT_0} = (1 - 1.5 \cdot L)\left[-a_1 - 2 \cdot a_2 \cdot (T - T_0) - 3 \cdot a_3 \cdot (T - T_0)^2 - 4 \cdot a_4 \cdot (T - T_0)^3\right]$$

$$\frac{dd}{dL} = 1.5\left[a_0 + a_1 \cdot (T - T_0) + a_2 \cdot (T - T_0)^2 + a_3 \cdot (T - T_0)^3 + a_4 \cdot (T - T_0)^4\right]$$

$$\frac{d\phi}{dT_0} = \frac{d\phi}{d\theta} \cdot \frac{d\theta}{dd} \cdot \frac{dd}{dT_0} = \frac{\cos(\alpha_0)}{r_{eff}} \cdot \frac{dd}{dT_0}$$

$$\frac{d\phi}{dL} = \frac{d\phi}{d\theta} \cdot \frac{d\theta}{dd} \cdot \frac{dd}{dL} = \frac{-1.5 \cdot d_p \cdot \cos(\alpha_0)}{r_{eff}}$$

$$\frac{d\phi}{d\phi_0} = 1$$

$$\frac{d\phi}{d\lambda_0} = 0$$

$$\frac{d\phi}{dh_0} = 0$$

$$\frac{d\phi}{d\alpha_0} = \theta \cdot \sin(\alpha_0)$$

The second column is composed of partials of longitude, λ:

$$\lambda = \lambda_0 - \operatorname{asin}\left(\frac{\sin(\theta) \cdot \sin(\alpha_0)}{\cos(\phi)}\right)$$

$$\frac{d\lambda}{d\theta} = \frac{1}{\sqrt{1 - \sin(\theta)^2 \cdot \frac{\sin(\alpha_0)^2}{\cos(\phi)^2}}} \cdot \cos(\theta) \cdot \frac{\sin(\alpha_0)}{\cos(\phi)} = \frac{\sin(\alpha_0)}{\cos(\phi)}$$

$$\frac{d\lambda}{dT_0} = \frac{d\lambda}{d\theta} \cdot \frac{d\theta}{dd} \cdot \frac{dd}{dT_0} = \frac{\sin(\alpha_0)}{r_{eff} \cdot \cos(\phi)} \cdot \frac{dd}{dT_0} = \frac{\sin(\alpha_0)}{r_{eff} \cdot \cos(\phi_0)} \cdot \frac{dd}{dT_0}$$

Assuming ϕ is approximately ϕ_0:

$$\frac{d\lambda}{dL} = \frac{d\lambda}{d\theta}\frac{d\theta}{dd}\frac{dd}{dL} = \frac{1.5 \cdot d_P \cdot \sin(\alpha_0)}{r_{eff}\cos(\phi)} = \frac{1.5 \cdot d_P \cdot \sin(\alpha_0)}{r_{eff}\cos(\phi_0)}$$

$$\frac{d\lambda}{d\phi_0} = \frac{1}{\sqrt{1 - \sin(\theta)^2 \frac{\sin(\alpha_0)^2}{\cos(\phi)^2}}} \cdot \sin(\theta) \cdot \frac{\sin(\alpha_0)}{\cos(\phi)^2} \cdot \sin(\phi) \cdot \frac{d\phi}{d\phi_0} = \sin(\theta) \cdot \sin(\alpha_0) \cdot \frac{\sin(\phi)}{\cos(\phi)^2} = 0$$

$$\frac{d\lambda}{d\lambda_0} = 1$$

$$\frac{d\lambda}{dh_0} = 0$$

$$\frac{d\lambda}{d\alpha_0} = \frac{1}{\sqrt{1 - \sin(\theta)^2 \frac{\sin(\alpha_0)^2}{\cos(\phi)^2}}} \cdot \sin(\theta) \cdot \frac{\cos(\alpha_0)}{\cos(\phi)} = \frac{\theta \cdot \cos(\alpha_0)}{\cos(\phi_0)}$$

And the third column elements are the partials of height, h:

$$h = h_0 + (1 + L) \cdot h_P$$

$$h_P = b_0 + b_1 \cdot T - T_0 + b_2 \cdot T - T_0^2 + b_3 \cdot T - T_0^3 + b_4 \cdot T - T_0^4$$

$$\frac{dh}{dT_0} = (1 + L) \cdot \left(b_1 - 2 \cdot b_2 \cdot T - T_0 - 3 \cdot b_3 \cdot T - T_0^2 - 4 \cdot b_4 \cdot T - T_0^3\right)$$

$$\frac{dh}{dL} = h_P$$

$$\frac{dh}{d\phi_0} = 0$$

$$\frac{dh}{d\lambda_0} = 0$$

$$\frac{dh}{dh_0} = 1$$

$$\frac{dh}{d\alpha}_0 = 0$$

$$\frac{dh}{dh}_0 = 1$$

The $\mathbf{A}_2$ matrix is the partials of (XYZ) with respect to (ϕ, λ, h):

$$x_{T_k} = \left[r_{local_k}\cos(\phi'_k) + alt_k\cos(\phi_k)\right]\cos(\lambda_k)$$

$$y_{T_k} = \left[r_{local_k}\cos(\phi'_k) + alt_k\cos(\phi_k)\right]\sin(\lambda_k)$$

$$z_{T_k} = r_{local_k}\sin(\phi'_k) + alt_k\sin(\phi_k)$$

Approximating r_{local} with r_{eff}, alt with h, ϕ' with ϕ, and taking the partials:

$$\frac{\partial x}{\partial \phi} = -(r_{eff} + h)\sin(\phi)\cos(\lambda)$$

$$\frac{\partial x}{\partial \lambda} = -(r_{eff} + h)\cos(\phi)\sin(\lambda)$$

$$\frac{\partial x}{\partial h} = \cos(\phi)\cos(\lambda)$$

$$\frac{\partial y}{\partial \phi} = -(r_{eff} - h)\sin(\phi)\sin(\lambda)$$

$$\frac{\partial y}{\partial \lambda} = (r_{eff} + h)\cos(\phi)\cos(\lambda)$$

$$\frac{\partial y}{\partial h} = \cos(\phi)\sin(\lambda)$$

$$\frac{\partial z}{\partial \phi} = (r_{eff} + h)\cos(\phi)$$

$$\frac{\partial z}{\partial \lambda} = 0$$

$$\frac{\partial z}{\partial h} = \sin(\phi)$$

APPENDIX B. QUALITATIVE ANALYSIS EQUATIONS AND PLOTS

The following equations and figures were composed with MATHCAD 5.0 Plus software. They are the lengthy equations referred to in Chapter 6 which were used to qualitatively analyze error effects upon LOS accuracy.

The LOS projection position equation in terms of the error sources is:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \begin{pmatrix} R\cdot\cos(gha)\cdot\cos(\delta) \\ R\cdot\sin(gha)\cdot\cos(\delta) \\ R\cdot\sin(\delta) \end{pmatrix} + \frac{r_{local}\cdot\sin\left(\eta + a\sin\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)}{\sin(\eta)} \begin{pmatrix} \cos(gha) & \sin(gha) & 0 \\ \sin(gha) & \cos(gha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\delta) & 0 & \sin(\delta) \\ 0 & 1 & 0 \\ \sin(\delta) & 0 & \cos(\delta) \end{pmatrix} \cdot \begin{pmatrix} -\cos(\eta) \\ \sin(\beta)\cdot\sin(\eta) \\ \cos(\beta)\cdot\sin(\eta) \end{pmatrix}$$

Expanded, this becomes:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \begin{bmatrix} R\cdot\cos(gha)\cdot\cos(\delta) - r_{local}\cdot\sin\left(\eta + a\sin\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\cos(gha)\cdot\cos(\delta)\cdot\cos(\eta) - \sin(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \cos(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)} \\ R\cdot\sin(gha)\cdot\cos(\delta) - r_{local}\cdot\sin\left(\eta + a\sin\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(gha)\cdot\cos(\delta)\cdot\cos(\eta) + \cos(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \sin(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)} \\ R\cdot\sin(\delta) - r_{local}\cdot\sin\left(\eta + a\sin\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(\delta)\cdot\cos(\eta) + \cos(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)} \end{bmatrix}$$

Breaking this down into each of its components:

$$x = R\cdot\cos(gha)\cdot\cos(\delta) - r_{local}\cdot\sin\left(\eta + a\sin\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\cos(gha)\cdot\cos(\delta)\cdot\cos(\eta) - \sin(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \cos(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$y = R\cdot\sin(gha)\cdot\cos(\delta) - r_{local}\cdot\sin\left(\eta + a\sin\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(gha)\cdot\cos(\delta)\cdot\cos(\eta) + \cos(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \sin(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$z = R\cdot\sin(\delta) - r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(\delta)\cdot\cos(\eta) + \cos(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

These coordinates can be differentiated with respect to each of the error sources. First, with respect to β:

$$\frac{dx}{d\beta} = -r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(-\sin(gha)\cdot\cos(\beta)\cdot\sin(\eta) + \cos(gha)\cdot\sin(\delta)\cdot\sin(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$\frac{dy}{d\beta} = -r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\cos(gha)\cdot\cos(\beta)\cdot\sin(\eta) + \sin(gha)\cdot\sin(\delta)\cdot\sin(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$\frac{dz}{d\beta} = r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\cos(\delta)\cdot\sin(\beta)$$

Next, with respect to η:

$$\frac{dx}{d\eta} = r_{local}\cdot\cos\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\left[1 + \frac{1}{\sqrt{1 - R^2\cdot\frac{\sin(\eta)^2}{r_{local}^2}}}\cdot R\cdot\frac{\cos(\eta)}{r_{local}}\right]\cdot\frac{(\cos(gha)\cdot\cos(\delta)\cdot\cos(\eta) - \sin(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \cos(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)} \ldots$$

$$+ \left[r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\right]\cdot\frac{(-\cos(gha)\cdot\cos(\delta)\cdot\sin(\eta) - \sin(gha)\cdot\sin(\beta)\cdot\cos(\eta) - \cos(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\cos(\eta))}{\sin(\eta)}\Bigg] \ldots$$

$$+ r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\cos(gha)\cdot\cos(\delta)\cdot\cos(\eta) - \sin(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \cos(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)^2}\cdot\cos(\eta)$$

$$\frac{dy}{d\eta} = r_{local}\cdot\cos\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\left[1 + \frac{1}{\sqrt{1 - R^2\cdot\frac{\sin(\eta)^2}{r_{local}^2}}}\cdot R\cdot\frac{\cos(\eta)}{r_{local}}\right]\cdot\frac{(\sin(gha)\cdot\cos(\delta)\cdot\cos(\eta) + \cos(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \sin(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)} \ldots$$

$$+ \left[-r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(-\sin(gha)\cdot\cos(\delta)\cdot\sin(\eta) + \cos(gha)\cdot\sin(\beta)\cos(\eta) - \sin(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\cos(\eta))}{\sin(\eta)}\right] \ldots$$

$$+ r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(gha)\cdot\cos(\delta)\cdot\cos(\eta) + \cos(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \sin(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)^2}\cdot\cos(\eta)$$

$$\frac{dz}{d\eta} = r_{local}\cdot\cos\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\left[1 + \frac{1}{\sqrt{1 - R^2\cdot\frac{\sin(\eta)^2}{r_{local}^2}}}\cdot R\cdot\frac{\cos(\eta)}{r_{local}}\right]\cdot\frac{(\sin(\delta)\cdot\cos(\eta) + \cos(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)} \ldots$$

$$+ \left[-r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(-\sin(\delta)\cdot\sin(\eta) + \cos(\delta)\cdot\cos(\beta)\cdot\cos(\eta))}{\sin(\eta)}\right] \ldots$$

$$+ r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(\delta)\cdot\cos(\eta) + \cos(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)^2}\cdot\cos(\eta)$$

Differentiating with respect to δ:

$$\frac{dx}{d\delta} = R\cdot\cos(gha)\cdot\sin(\delta) \quad r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\cos(gha)\cdot\sin(\delta)\cdot\cos(\eta) - \cos(gha)\cdot\cos(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$\frac{dy}{d\delta} = R\cdot\sin(gha)\cdot\sin(\delta) - r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(gha)\cdot\sin(\delta)\cdot\cos(\eta) - \sin(gha)\cdot\cos(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$\frac{dz}{d\delta} = R\cdot\cos(\delta) - r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\cos(\delta)\cdot\cos(\eta) - \sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

Differentiating with respect to R:

$$\frac{dx}{dR} = \cos(gha)\cdot\cos(\delta) - \frac{\cos\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)}{\sqrt{1 - R^2\cdot\frac{\sin(\eta)^2}{r_{local}^2}}}\cdot(\cos(gha)\cdot\cos(\delta)\cdot\cos(\eta) - \sin(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \cos(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))$$

$$\frac{dy}{dR} = \sin(gha)\cdot\cos(\delta) - \frac{\cos\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)}{\sqrt{1 - R^2\cdot\frac{\sin(\eta)^2}{r_{local}^2}}}\cdot(\sin(gha)\cdot\cos(\delta)\cdot\cos(\eta) + \cos(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \sin(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))$$

$$\frac{dz}{dR} = \sin(\delta) - \frac{\cos\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)}{\sqrt{1 - R^2\cdot\frac{\sin(\eta)^2}{r_{local}^2}}}\cdot(\sin(\delta)\cdot\cos(\eta) + \cos(\delta)\cdot\cos(\beta)\cdot\sin(\eta))$$

And last, with respect to gha:

$$\frac{dx}{dgha} = R\cdot\sin(gha)\cdot\cos(\delta) - r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\sin(gha)\cdot\cos(\delta)\cdot\cos(\eta) - \cos(gha)\cdot\sin(\beta)\cdot\sin(\eta) + \sin(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$\frac{dy}{dgha} = R\cdot\cos(gha)\cdot\cos(\delta) - r_{local}\cdot\sin\left(\eta + \operatorname{asin}\left(R\cdot\frac{\sin(\eta)}{r_{local}}\right)\right)\cdot\frac{(\cos(gha)\cdot\cos(\delta)\cdot\cos(\eta) - \sin(gha)\cdot\sin(\beta)\cdot\sin(\eta) - \cos(gha)\cdot\sin(\delta)\cdot\cos(\beta)\cdot\sin(\eta))}{\sin(\eta)}$$

$$\frac{dz}{dgha} = 0$$

The geocentric longitude and latitude can be expressed in terms of (XYZ) coordinates:

$$\lambda = \operatorname{atan}\left(\frac{y}{x}\right)$$

$$\frac{d\lambda}{dx} = \frac{-y}{\left[x^2 \cdot \left(1 + \frac{y^2}{x^2}\right)\right]}$$

$$\frac{d\lambda}{dy} = \frac{1}{\left[x \cdot \left(1 + \frac{y^2}{x^2}\right)\right]}$$

$$\phi' = \operatorname{atan}\left(\frac{z}{\sqrt{x^2 + y^2}}\right)$$

$$\frac{d\phi'}{dx} = \frac{-z}{\left(x^2 + y^2\right)^{\frac{3}{2}} \cdot \left[1 + \frac{z^2}{x^2 + y^2}\right]} \cdot x$$

$$\frac{d\phi'}{dy} = \frac{-z}{\left(x^2 + y^2\right)^{\frac{3}{2}} \cdot \left[1 + \frac{z^2}{x^2 + y^2}\right]} \cdot y$$

$$\frac{d\phi'}{dz} = \frac{1}{\sqrt{x^2 + y^2} \cdot \left[1 + \frac{z^2}{x^2 + y^2}\right]}$$

The differentials of the geocentric longitude and latitude can be determined using the Chain Rule. Using longitude and satellite radius as example quantities:

$$\frac{d\lambda}{dR} = \frac{d\lambda}{dx} \cdot \frac{dx}{dR} + \frac{d\lambda}{dy} \cdot \frac{dy}{dR}$$

Substituting the following quantities into the previous equations:

$$R = 42164.17\text{km}$$

$$gha = 70 \cdot \text{deg}$$

$$\delta = 0$$

$$r_{local} = 6378.137\text{km}$$

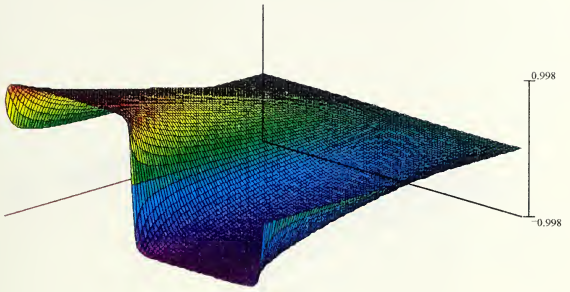
η = varies 0° to 8.5°

β = varies 0° to 360°

This gives expressions for the variations of the geocentric longitude and latitude due to errors in LOS projections from a DSP satellite with a Greenwich hour angle of 70°. These expressions are transformed into plots for examination.

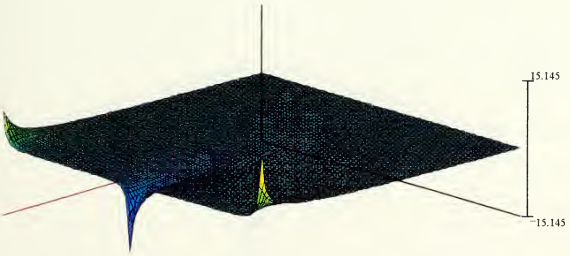
The following plots show the differentials of geocentric longitude and latitude with respect to the five error sources. The horizontal axis pointing to the left is η, the horizontal axis pointing toward the right is β, and the vertical axis is the differential evaluated at each (η, β) coordinate.

$$d\phi'd\beta_{i,j} = d\phi'dx_{i,j}\cdot dx d\beta_{i,j} + d\phi'dy_{i,j}\cdot dy d\beta_{i,j} + d\phi'dz_{i,j}\cdot dz d\beta_{i,j}$$



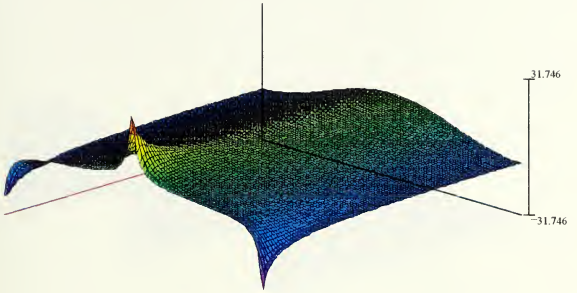
$d\phi'd\beta$

$$d\lambda d\beta_{i,j} = d\lambda dx_{i,j}\cdot dx d\beta_{i,j} + d\lambda dy_{i,j}\cdot dy d\beta_{i,j}$$



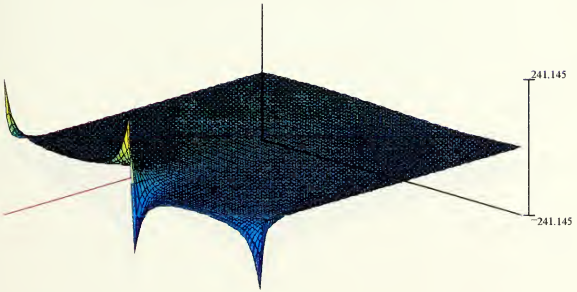
$d\lambda d\beta$

$$d\phi'd\eta_{i,j} = d\phi'dx_{i,j} \cdot dx d\eta_{i,j} + d\phi'dy_{i,j} \cdot dy d\eta_{i,j} + d\phi'dz_{i,j} \cdot dz d\eta_{i,j}$$



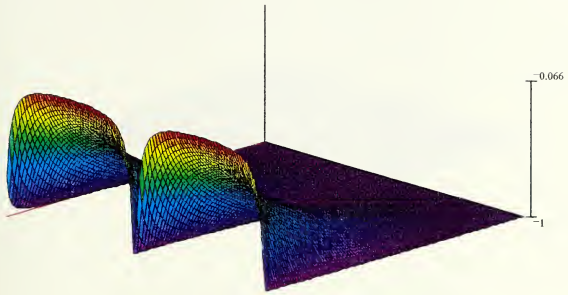
$d\phi'd\eta$

$$d\lambda d\eta_{i,j} = d\lambda dx_{i,j} \cdot dx d\eta_{i,j} + d\lambda dy_{i,j} \cdot dy d\eta_{i,j}$$



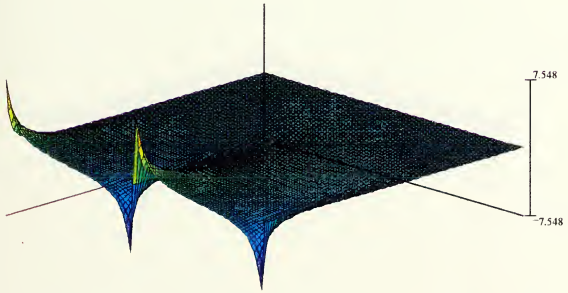
$d\lambda d\eta$

$$d\phi'd\delta_{i,j} = d\phi'dx_{i,j} \cdot dx d\delta_{i,j} + d\phi'dy_{i,j} \cdot dy d\delta_{i,j} + d\phi'dz_{i,j} \cdot dz d\delta_{i,j}$$



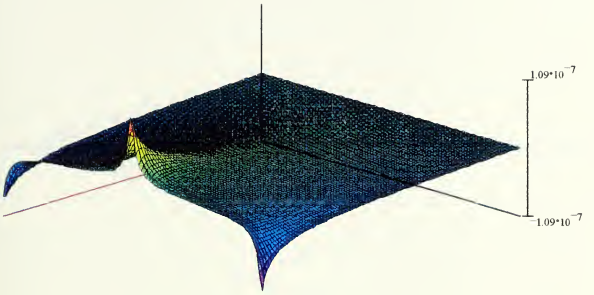
$d\phi'd\delta$

$$d\lambda d\delta_{i,j} = d\lambda dx_{i,j} \cdot dx d\delta_{i,j} + d\lambda dy_{i,j} \cdot dy d\delta_{i,j}$$



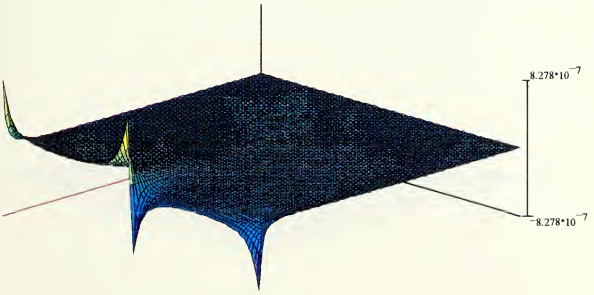
$d\lambda d\delta$

$$d\phi'dR_{i,j} = d\phi'dx_{i,j} \cdot dxdR_{i,j} + d\phi'dy_{i,j} \cdot dydR_{i,j} + d\phi'dz_{i,j} \cdot dzdR_{i,j}$$



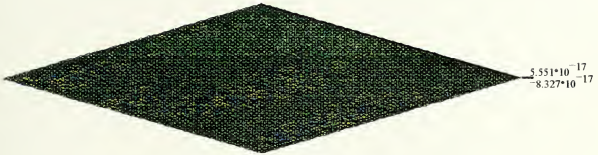
$d\phi'dR$

$$d\lambda dR_{i,j} = d\lambda dx_{i,j} \cdot dxdR_{i,j} + d\lambda dy_{i,j} \cdot dydR_{i,j}$$



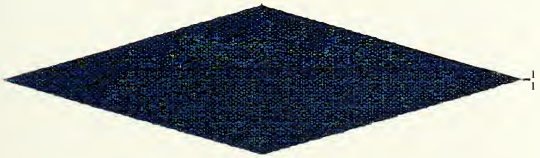
$d\lambda dR$

$$d\phi dgha_{i,j} = d\phi dx_{i,j} \cdot dx dgha_{i,j} + d\phi dy_{i,j} \cdot dy dgha_{i,j}$$



dϕdgha

$$d\lambda dgha_{i,j} = d\lambda dx_{i,j} \cdot dx dgha_{i,j} + d\lambda dy_{i,j} \cdot dy dgha_{i,j}$$



dλdgha

APPENDIX C. MATLAB CODE

The following is the MATLAB™ program used to simulate the TALON SHIELD / ALERT state vector estimation algorithm. It is commented in blue font.

```
%                Martin Beaulieu
%             DSP Estimation Algorithm

% Initialize variables

clear;                          % clear memory
format long;                    % use 15-digit numerical format
numpoints=1000;                 % number of data points to run
reff=6371;                      % spherical earth radius (km)
f=1/298.257;                    % WGS-84 flattening factor for oblate earth
re=6378.137;                    % oblate earth equatorial radius (km)
a0=0.0749291380897402,          % coefficients in range profile equation
a1=-0.0501647424642791;
a2=0.0041947951949387,
a3=-0.0000141392516462238;
a4=8.51293969732759E-07;
b0=0.884282320240989,           % coefficients in height profile equation
b1=-0.129163814041924;
b2=0.0137288843548327;
b3=-0.000159307288768991,
b4=1.36938557528199E-06;
tmax=62.5;                      % TBM maximum motor burn time (seconds)
mu=398601.2,                    % earth gravitational constant (km^3 sec^2)
d2r=pi/180;                     % degrees to radians
r2d=180/pi;                     % radians to degrees
we=15*d2r/3600;                 % earth rotation rate (15 degrees/hour)

%----------------------------------------------------------------------------------------

% This section generates the observation table (time, az, el, gha, dec, and radius).  The first
% step is to choose the tactical parameters, then calculate how that missile profile would
% appear to an orbiting sensor.

% Initial tactical parameters

for ii=0:11                     % loop through 12 headings around compass rose
  T0=100;                       % launch time of day in seconds
  L=0;                          % loft parameter
  %lat0=12.783*d2r;lon0=45.050*d2r;   % Aden, Yemen
```

```
lat0=33.333*d2r;lon0=44.433*d2r,          % Baghdad. Iraq
%lat0=33.500*d2r;lon0=36.319*d2r,         % Damascus, Syria
%lat0=24.633*d2r;lon0=46.717*d2r,         % Riyadh. Saudi Arabia
%lat0=35.666*d2r;lon0=51.433*d2r;         % Tehran. Iran
h0=0;                                     % launch altitude
hdg0=ii*pi/6;                             % launch heading in 30 degree steps
data=[T0 L lon0 lat0 h0 hdg0 zeros(1,9)]; % true tactical parameters

% Generate observation time matrix

obs1=T0+30+5*randn(1);                    % first observation time - clouds. vapor. etc
dt1=10*rand(1);dt2=10*rand(1);            % two random time intervals. 0<dt<10 sec
ttime=[obs1;obs1+dt1;obs1+dt2];           % first 3 observation times
ttime=sort(ttime);                        % put times in chronological order
nextobs=obs1+10;                          % next sequential observation
j=1;                                      % counting index
while nextobs <= tmax+T0,                 % if next observation is during motor burn
  ttime=[ttime;ttime(j)+10];              % add next observation to time matrix
  j=j+1;                                  % increment counter
  nextobs=ttime(j)+10;                    % next sequential observation
end                                       % loop until matrix is completed
n=length(ttime);                          % number of observations

% Generate satellite position matrix

gha=[10*d2r;70*d2r;105*d2r],              % Greenwich hour angle
satord=ceil(6*rand(1));                   %  randomly select the order in
if satord == 1                            %  which satellites observe TBM
  tgha=[gha(1);gha(2);gha(3)],
elseif satord == 2
  tgha=[gha(1);gha(3);gha(2)],
elseif satord == 3
  tgha=[gha(2);gha(1);gha(3)],
elseif satord == 4
  tgha=[gha(2);gha(3);gha(1)],
elseif satord == 5
  tgha=[gha(3);gha(1);gha(2)],
elseif satord == 6
  tgha=[gha(3);gha(2);gha(1)],
end
tdec=[0;0;0];                             % declination
tradius=[42164.17;42164.17;42164.17],     % geosynchronous radius
for i=4:n                                 %  make satpos matrix the same size as time
  tgha=[tgha;tgha(i-3)],
  tdec=[tdec;tdec(i-3)],
```

```
  tradius=[tradius;tradius(i-3,:)];
end
satpos=[tradius.*cos(tdec).*cos(tgha)
        tradius.*cos(tdec).*sin(tgha)
        tradius.*sin(tdec)];                    % satellite position
```

% Apply profile to observation times:

```
t=ttime-T0;                                     % time of flight
dp=a0+a1*t+a2*t.^2+a3*t.^3+a4*t.^4;             % nominal profile distance
d=(1-1.5*L)*dp;                                 % loft-modified distance
hp=b0+b1*t+b2*t.^2+b3*t.^3+b4*t.^4;             % nominal profile height
h=(1+L)*hp;                                     % loft-modified height
```

% Generate target position:

```
theta=d/reff;                                   % earth-central angle
lati=pi/2-acos(cos(theta)*sin(lat0)+sin(theta)*cos(lat0)*cos(hdg0));% latitude
long=lon0+asin(sin(theta)*sin(hdg0)./cos(lati));% longitude
alt=h0+h;                                       % altitude
gcl=atan((1-f)^2*tan(lati));                    % geocentric latitude
rloc=re*(1-f)./sqrt((1-f)^2*cos(gcl).^2+sin(gcl).^2);% local radius
tgtpos=[(rloc.*cos(gcl)+alt.*cos(lati)).*cos(long)
        (rloc.*cos(gcl)+alt.*cos(lati)).*sin(long)
         rloc.*sin(gcl)+alt.*sin(lati)];        % target position
```

% Generate line-of-sight vector and transform into focal plane coordinates

```
delta=tgtpos-satpos;                            % line-of-sight vector
r3=[];r3t=[];UEN=[];                            % reset variables
for i=1:n
  rot1=[cos(tgha(i)) -sin(tgha(i)) 0;sin(tgha(i)) cos(tgha(i)) 0;0 0 1];
  rot2=[cos(tdec(i)) 0 -sin(tdec(i));0 1 0;sin(tdec(i)) 0 cos(tdec(i))];
  rot3=rot1*rot2;
  r3t=[r3t;rot3'];
  r3=[r3;rot3];
  UEN(i,:)=(r3t(3*i-2:3*i,:)*delta(i,:)')';     % Up-East-North coordinates
end
tbeta=rem(-pi/2-atan2(UEN(:,3),UEN(:,2))+(2*pi),(2*pi));% true azimuth
teta=atan(sqrt(UEN(:,2).^2+UEN(:,3).^2)./(-UEN(:,1)));  % true elevation
```

%----------

% Using generated data (ttime, tbeta, teta, tgha, tdec, tradius),
% compute tactical parameters using least-squares iteration method

```
for index=1:numpoints;
  index=round(index);                          % ensure index is an integer

% Initialize variables

  clear time radius gha dec beta eta T0 L lat0 lon0 h0 alt0 hdg0 tp;
  sse=1;dsse=1;sse1=1;failindex=0;r3t=[];r3=[];   % reset variables
  errtime=zeros(n,1);                          % time error
  errrad=zeros(n,1);                           % satellite radius error
  errgha=zeros(n,1);                           % satellite gha error
  errdec=zeros(n,1);                           % satellite dec error
  errbeta=zeros(n,1);                          % azimuth error
  erreta=zeros(n,1);                           % elevation error

  % Change time, beta, eta, and satpos by some error, epsilon

  if index > 1
    errtime=1e-3*(rand(n,1)-0.5);              % <=+-0.001 second
    errrad=0.2*randn(3,1);                     % variance of +-200 meters
    errgha=atan(0.2/42164.17)*randn(3,1);      % variance of +-200 meters
    errdec=atan(0.2/42164.17)*randn(3,1);      % variance of +-200 meters
    for i=4:n                                  %      fix satellite positions
        errrad=[errrad;errrad(i-3)];           %      during single run
        errgha=[errgha;errgha(i-3)],
        errdec=[errdec;errdec(i-3)],
    end
    errbeta=5e-6*randn(n,1);                   % variance= +-5microradians
    erreta=5e-6*randn(n,1);                    % variance= +-5microradians
  end
  time=ttime+errtime;                          % time = truth + error
  radius=tradius+errrad,                       % radius = truth + error
  gha=tgha+errgha;                             % gha = truth + error
  dec=tdec+errdec;                             % dec = truth + error
  beta=tbeta+errbeta;                          % beta = truth + error
  eta=teta+erreta;                             % eta = truth + error

  % Compute target position.

  satpos=[radius.*cos(dec).*cos(gha)
          radius.*cos(dec).*sin(gha)
          radius.*sin(dec)];                   % satellite position
  for i=1:n                                    % rotation matrices
    rot1=[cos(gha(i)) -sin(gha(i)) 0;sin(gha(i)) cos(gha(i)) 0;0 0 1];
    rot2=[cos(dec(i)) 0 -sin(dec(i));0 1 0;sin(dec(i)) 0 cos(dec(i))];
```

```
  rot3=rot1*rot2;
  r3t=[r3t;rot3'];
  r3=[r3;rot3];
  uen=[-cos(eta(i)),
       -sin(beta(i))*sin(eta(i)),
       -cos(beta(i))*sin(eta(i))];          % Up-East -North coordinates
  ehat=rot3*uen;                            % line of sight direction
  chi=pi-asin((norm(satpos(i,:))*sin(eta)/reff));  % computation of length
  rho=reff/sin(eta)*sin(eta+chi);           % for los vector
  los=rho*ehat;                             % line-of-sight vector
  tgtpos=los+satpos(i,:)';                  % target position
  lat(i)=atan(tgtpos(3)/sqrt(tgtpos(1)^2+tgtpos(2)^2));   % latitude
  lon(i)=atan2(tgtpos(2),tgtpos(1));                      % longitude
end

% Intial estimate of tactical parameters

lat0=(lat(1)+lat(2)+lat(3))/3;              % first obs latitude
lon0=(lon(1)+lon(2)+lon(3))/3;              % first obs longitude
latbo=(lat(n-2)+lat(n-1)+lat(n))/3;         % latitude at last obs
lonbo=(lon(n-2)+lon(n-1)+lon(n))/3,         % longitude at last obs
hdg0=rem(pi/2-atan2(latbo-lat0,(lonbo-lon0)*cos(lat0))+(2*pi),(2*pi)); % aInch heading
T0=time(1)-20,                              % launch time
L=0;                                        % loft parameter
h0=0;                                       % launch height above WGS-84 ellipsoid
tp=[T0;L;lat0;lon0;h0;hdg0];                % tactical parameters matrix

% Iterate on initial estimates until the difference between the sum of the
% squares of the errors between two consecutive runs is <=10*eps:

while dsse > 10*eps                         % loop until dsse <= 10*eps
  duv=[];A=[];                              % reset variables
  T0=tp(1);L=tp(2),lat0=tp(3);              % update tp matrix values
  lon0=tp(4);h0=tp(5);hdg0=rem(tp(6),2*pi); % with dtp's added
  t=time-T0;                                % time-of-flight
  dp=a0+a1*t+a2*t.^2+a3*t.^3+a4*t.^4;       % nominal profile distance
  d=(1-1.5*L)*dp;                           % loft-modified distance
  hp=b0+b1*t+b2*t.^2+b3*t.^3+b4*t.^4;       % nominal profile height
  h=(1+L)*hp;                               % loft-modified height
  theta=d/reff;                             % earth-central angle
  lati=pi/2-acos(cos(theta)*sin(lat0)+sin(theta)*cos(lat0)*cos(hdg0));% geodtic latitude
  long=lon0+asin(sin(theta)*sin(hdg0)./cos(lati)); % longitude
  alt=h0+h;                                 % altitude
  gcl=atan((1-f)^2*tan(lati));              % geocentric latitude
  rloc=re*(1-f)./sqrt((1-f)^2*cos(gcl).^2+sin(gcl).^2);% local radius
```

```
tgtpos=[(rloc.*cos(gcl)+alt.*cos(lati)).*cos(long)
        (rloc.*cos(gcl)+alt.*cos(lati)).*sin(long)
         rloc.*sin(gcl)+alt.*sin(lati)];          % target position
delta=tgtpos-satpos;                              % line-of-sight vector
for i=1:n                                         % rotate into UEN coordinates
  UEN(i,:)=(r3t(3*i-2:3*i,:)*delta(i,:)')';
end
uhat=-UEN(:,2)./UEN(:,1);                         %  estimated focal
vhat=-UEN(:,3)./UEN(:,1);                         %  plane coordinates
u=-tan(eta).*sin(beta);                           %     actual focal
v=-tan(eta).*cos(beta);                           %     plane coordinates
du=u-uhat;                                        %        difference between
dv=v-vhat;                                        %        estimate and actual

% Compute A matrix:

c1=cos(hdg0)/reff;c2=sin(hdg0)/reff/cos(lat0);    % constants in A matrix
c3=sin(hdg0)/reff;c4=cos(hdg0)/reff/cos(lat0),
dddT0=-(1-1.5*L)*(a1+2*a2*t+3*a3*t.^2+4*a4*t.^3); % -horizontal speed
dhdT0=-(1+L)*(b1+2*b2*t+3*b3*t.^2+4*b4*t.^3);     % -vertical speed
dudUEN=[UEN(:,2)./UEN(:,1).^2 -1./UEN(:,1) zeros(size(time))];
dvdUEN=[UEN(:,3)./UEN(:,1).^2 zeros(size(time)) -1./UEN(:,1)];
% change in focal plane coordinates wrt changes in UEN coordinates
for i=1:n                                         % construct A matrix
  duv=[duv;du(i);dv(i)];                          % error values matrix
  A1= [dddT0(i)*c1 dddT0(i)*c2 dhdT0(i);
       -1.5*dp(i)*c1 -1.5*dp(i)*c2 hp(i);
       1 0 0;0 1 0;0 0 1;-d(i)*c3 d(i)*c4 0];
  A2=[-(rloc(i)+alt(i))*sin(lati(i))*cos(long(i))
      -(rloc(i)+alt(i))*sin(lati(i))*sin(long(i))
       (rloc(i)+alt(i))*cos(lati(i));
      -(rloc(i)+alt(i))*cos(lati(i))*sin(long(i))
       (rloc(i)+alt(i))*cos(lati(i))*cos(long(i)) 0;
       cos(lati(i))*cos(long(i)) cos(lati(i))*sin(long(i)) sin(lati(i))];
  Atmp=A1*A2*r3(3*i-2:3*i,:);                     % A3=(delta UEN / delta xyz)=r3
  A=[A;(Atmp*dudUEN(i,:)')';(Atmp*dvdUEN(i,:)')'];          % total A matrix
end
AT=pinv(A);                                       % find pseudoinverse of A
dtp=AT*duv;                                       % find changes to tp
tp=tp+dtp;                                        % add changes to tp
sse1=sum(duv.^2);                                 % find sum of the squares of the error
dsse=abs(sse-sse1);                               % difference between iterations
if sse1 < sse
    sse=sse1;
end
```

```
  failindex=failindex+1;                    % increment counter
  if failindex>100                          % if statement to prevent continuous looping
     dsse=0;                                % in the event that it does not converge
  end
end                                         % loop to next tp iteration

% Burnout time estimation

Tlast=time(n);                              % last observation time
Tnext=2*time(n-2)-time(n-5),                % next observation if TBM was still burning
Tmax=T0+tmax;                               % max observation time from profile
if index ==1
  Tbo=T0+tmax;
elseif Tmax>=Tnext                          % pick burn out time as half of the time
  Tbo=Tlast+0.5*(Tnext-Tlast),              % between Tlast and Tnext or Tlast and
else                                        % Tmax depending on relative
  Tbo=Tlast+0.5*(Tmax-Tlast),               % magnitude of Tmax and Tnext
end
tbo=Tbo-T0;                                 % burnout time-of-flight (close to tmax)

% Calculates state vector at burnout

d=(1-1.5*L)*(a0+a1*tbo+a2*tbo^2+a3*tbo^3+a4*tbo^4);          % distance
ddot=(1-1.5*L)*(a1+2*a2*tbo+3*a3*tbo^2+4*a4*tbo^3);          % horizontal speed
h=(1+L)*(b0+b1*tbo+b2*tbo^2+b3*tbo^3+b4*tbo^4);              % height
hdot=(1+L)*(b1+2*b2*tbo+3*b3*tbo^2+4*b4*tbo^3),              % vertical speed
theta=d/reff;                                                % earth-central angle
latbo=pi/2-acos(cos(theta)*sin(lat0)+sin(theta)*cos(lat0)*cos(hdg0));% geodtic latitude
lonbo=lon0+asin(sin(theta)*sin(hdg0)/cos(latbo)),            % burnout longitude
hbo=h0+h;                                                    % burnout altitude
Vbo=sqrt(ddot^2+hdot^2);                                     % burnout velocity
fpabo=atan(hdot/ddot);                                       % flight path angle
hdgbo=rem(asin(cos(lat0)*sin(hdg0)/cos(latbo))+(2*pi),(2*pi)); % burnout heading

% Calculates ballistic trajectory of target to impact

altbo=h0+hbo;                                                % burnout altitude
gclbo=atan(((1-f)^2)*tan(latbo)),                            % geocentric latitude
rlocbo=re*(1-f)/sqrt((1-f)^2*cos(gclbo)^2+sin(gclbo)^2);     % local radius
tgtposbo=[(rlocbo*cos(gclbo)+altbo*cos(latbo))*cos(lonbo)
          (rlocbo*cos(gclbo)+altbo*cos(latbo))*sin(lonbo)
           rlocbo*sin(gclbo)+altbo*sin(latbo)];  % TBM position at burnout
rbo=norm(tgtposbo);                          % radius at burnout
vo=rlocbo*we*cos(gclbo);                     % initial inertial velocity
Vu=Vbo*sin(fpabo);                           % up component of velocity
```

```
Ve=Vbo*cos(fpabo)*sin(hdgbo)+vo,          % east component of velocity
Vn=Vbo*cos(fpabo)*cos(hdgbo);            % north component of velocity
Vin=sqrt(Vu.^2+Ve.^2+Vn.^2);             % inertial speed
fpain=asin(Vu/Vin);                      % inertial flight path angle
hdgin=rem(pi/2-atan2(-Ve,Vin)+2*pi,2*pi); % inertial heading
Qin=Vin^2*rbo/mu;                        % energy parameter
ein=sqrt(1+Qin*(Qin-2)*cos(fpain)^2);    % eccentricity of orbit
PSI=2*acos((1-Qin*cos(fpain)^2)/ein)+theta; % freeflight eca
E=acos((ein-cos(PSI/2))/(1-ein*cos(PSI/2)));  % eccentric anomaly
ain=rbo/(2-Qin);                         % semimajor axis
tff=2*sqrt(ain^3/mu)*(pi-E+ein*sin(E));  % time of free flight
gclim=asin(sin(gclbo)*cos(PSI)+cos(gclbo)*sin(PSI)*cos(hdgin)); %geocentric latitude
latim=atan(tan(gclim)/(1-f)^2);          % geodetic latitude of impact
fflon=acos((cos(PSI)-sin(gclim)*sin(gclbo))/(cos(gclim)*cos(gclbo)))-we*tff,
lonim=lonbo+fflon;                       % longitude of impact
tim=2*tbo+tff;                           % total flight time
Tim=T0+tim,                              % time of day of impact
 data=[data;tp(1) tp(2) tp(4) tp(3) tp(5) tp(6) lonbo latbo hbo fpabo hdgbo Vbo Tim
lonim latim];                            % save data in matrix
end                                      % end of numpoint loop

% Plotting routines and data analysis

tllon=data(2,3)*r2d;                     % true launch longitude
tllat=data(2,4)*r2d;                     % true launch latitude
llon=data(3:index+1,3)*r2d,              % corrupt launch longitude
llat=data(3:index+1,4)*r2d;              % corrupt launch latitude
mllon=mean(data(3:index+1,3))*r2d;       % mean of corrupt longitudes
mllat=mean(data(3:index+1,4))*r2d,       % mean of corrupt latitudes
dllon=tllon-mllon;                       % diff between mean and true llon
dllat=tllat-mllat;                       % diff between mean and true llat

tblon=data(2,7)*r2d,                     % true burnout longitude
tblat=data(2,8)*r2d;                     % true burnout latitude
tbalt=data(2,9);                         % true burnout altitude
blon=data(3:index+1,7)*r2d,              % corrupt burnout longitude
blat=data(3:index+1,8)*r2d,              % corrupt burnout latitude
balt=data(3:index+1,9);                  % corrupt burnout altitude
mblon=mean(data(3:index+1,7))*r2d;       % mean of corrupt bo longitudes
mblat=mean(data(3:index+1,8))*r2d;       % mean of corrupt bo latitudes
mbalt=mean(data(3:index+1,9));           % mean of corrupt bo altitudes
dblon=tblon-mblon;                       % diff between mean and true blon
dblat=tblat-mblat;                       % diff between mean and true blat
dbalt=tbalt-mbalt;                       % diff between mean and true blat
```

```
tilon=data(2,14)*r2d;                    % true impact longitude
tilat=data(2,15)*r2d,                    % true impact latitude
ilon=data(3:index+1,14)*r2d,             % corrupt impact longitude
ilat=data(3:index+1,15)*r2d;             % corrupt impact latitude
milon=mean(data(3:index+1,14))*r2d;      % mean of corrupt im longitudes
milat=mean(data(3:index+1,15))*r2d;      % mean of corrupt im latitudes
dilon=tilon-milon;                       % diff between mean and true ilon
dilat=tilat-milat;                       % diff between mean and true ilat

lpts=[llon-tllon llat-tllat];            % center data about launch truth
lk=cov(lpts);                            % covariance of scatter
[lv,ld]=eig(lk),                         % eigen value & vector of covariance
lab=2*sqrt(diag(ld));                    % dimensions of ellipse
la=lab(1);                               % launch ellipse semimajor axis
lb=lab(2),                               % launch ellipse semiminor axis
lth=pi/2-atan(lv(1,1)/lv(2,1));          % angle to rotate ellipse

bpts=[blon-tblon blat-tblat balt-tbalt];  % center data about burnout truth
[bv1,bd1]=eig(cov(bpts(:,1),bpts(:,2)));  % eigen val and vector of covariance
babc=2*sqrt(diag(bd1));                  % dimensions of ellipse
ba=babc(1),bb=babc(2),bc=babc(3),        % burnout ellipse semimajor axis

ipts=[ilon-tilon ilat-tilat],            % center data about launch truth
ik=cov(ipts);                            % covariance of scatter
[iv,id]=eig(ik),                         % eigen value & vector of covarance
iab=2*sqrt(diag(id));                    % dimensions of ellipse
ia=iab(1);                               % impact ellipse semimajor axis
ib=iab(2),                               % impact ellipse semiminor axis
ith=pi/2-atan(iv(1,1)/iv(2,1));          % angle to rotate ellipse

% Save data to analyze

 data=[data;la lb lth ba1 bb1 bth1 ba2 bb2 bth2 ba3 bb3 bth3 ia ib ith; mllon mllat mblon
mblat mbalt milon milat dllon dllat dblon dblat dbalt dilon dilat 0];
 if ii==0
   save b0t data -ascii -double
 elseif ii==1
   save b3t data -ascii -double
 elseif ii==2
   save b6t data -ascii -double
 elseif ii==3
   save b9t data -ascii -double
 elseif ii==4
   save b12t data -ascii -double
 elseif ii==5
```

```
    save b15t data -ascii -double
  elseif ii==6
    save b18t data -ascii -double
  elseif ii==7
    save b21t data -ascii -double
  elseif ii==8
    save b24t data -ascii -double
  elseif ii==9
    save bt27 data -ascii -double
  elseif ii==10
    save bt30 data -ascii -double
  elseif ii==11
    save bt33 data -ascii -double
  end
  clear data
end
```

LIST OF REFERENCES

1. Rodrigues, Louis J., "Defense Support Program Ground Station Upgrades Not Based on Validated Requirements", *GAO Report to the Acting Secretary of the Air Force*, United States General Accounting Office, May 1993.

2. Bontrager, Mark D., "Defense Support Program," *Space Operations Orientation Course Handbook*, Air Force Space Command, 1993, p. 179-190.

3. Jerardi, Thomas, *Ideal DSP System*, unpublished paper, Johns Hopkins University / Applied Physics Laboratory, 1994.

4. Bishop, Arthur, DSP Sensor Evolutionary Development (SED) Familiarization Manual, Aerojet ElectroSystems, 1979.

5. Jerardi, Thomas, *TALON SHIELD ALERT State Vector Estimation*, unpublished paper, JHUAPL, 1995.

6. Bate, Roger R., et al, *Fundamentals of Astrodynamics*, Dover Publications, Inc., 1971.

7. Torrieri, Don J., "Statistical Theory of Passive Location Systems," *IEEE Transactionson Aerospace and Electronic Systems*, volume AES-20, number 2, March 1994, p. 183-198

BIBLIOGRAHPY

Bate, Roger R., et al., *Fundamentals of Astrodynamics*, Dover Publications, Inc., 1971.

Bishop, Arthur, *DSP Sensor Evolutionary Development (SED) Familiarization Manual*, Aerojet ElectroSystems, 1979.

Bontrager, Mark D., "Defense Support Program," *Space Operations Orientation Course Handbook*, Air Force Space Command, 1993, p. 179-190.

Erilane, Robert, "Satellite Tracking Accuracy," presentation slides, POET/The Aerospace Corporation, December 6, 1994.

Finkleman, David, "The Effect Of Satellite Uncertainties Upon The Location Of Perceived Events," unpublished paper, USSPACECOM, 1993.

Nakano, Hiroshi, *DSP-1 Attitude Control Subsystem, Parts 1-2*, TRW Military Space Systems Division, 1993.

Jerardi, Thomas, "TALON SHIELD/ALERT State Vector Estimation," unpublished paper, JHUAPL, 1995.

Jerardi, Thomas, "Ideal DSP System", unpublished paper, JHUAPL, 1994.

Jerardi, Thomas, "Error Model for Tactical DSP", unpublished paper, JHUAPL, 1994.

Jerardi, Thomas, "Error Analysis of Stereo DSP", unpublished paper, JHUAPL, 1993.

Leite, Diane, "Space Links Enhance Missile Defense", Space Tracks, Naval Space Command, Spring, 1995, p. 7-8.

Perrella, Albert J., Jr., "Estimating TBM Parameters From Space Based Observations During Boost Phase", Technical Note Draft, POET, 1995.

Perrella, Albert J., Jr., "POET Cueing Cookbook", Presentation to the TMD COEA Cueing Workshop, January 12-13, 1995.

Rodrgues, Louis J., "Defense Support Program Ground Station Upgrades Not Based on Validated Requirements," *Report to the Acting Secretary of the Air* Force, United States General Accounting Office, 1993.

"Tactical Surveillance Detachments Move", *Space Tracks*, Naval Space Command, Spring, 1995, p. 7-8.

Theater Missile Defense: Systems and Issues - 1994, A Collection of Papers, American Institute of Aeronautics and Astronautics, 1994.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Road, STE 0944
   Fort Belvoir VA 22060-6218

2. Library, Code 13 2
   Naval Postgraduate School
   Monterey CA 93943-5101

3. Dr. D. J. Collins, Code AA 1
   Naval Postgraduate School
   Monterey CA 93943

4. Dr. Terry Alfriend, Code Sp/Al 2
   Naval Postgraduate School
   Monterey CA 93943

5. Thomas Jerardi 1
   JHU/APL
   Johns Hopkins Road
   Laurel MD 20723-6099

6. Major Tom Harvill, USA 1
   ATTN: CSSD-PI-A
   P.O. Box 1500
   Huntsville AL 35807-3801

7. Lieutenant Colonel Marty Remedes 1
   TALON SHIELD
   720 Erwin Avenue, STE 2306
   Falcon AFB CO 80912-7200

8. Captain Kevin Giammo 1
   50SW / CCX
   300 O'Malley Avenue, STE 20
   Falcon AFB CO 80912-3020

9. Captain Legrow 2
   Department of the Navy, CNO
   N632, Room 5P773
   Pentagon
   Washington DC 20350-2000

10. Albert W. Bevan, Jr., Ph. D. 1
SWC/AET
730 Irwin Avenue, Suite 83
Falcon AFB CO 80912-7383

11. Martin Beaulieu 2
183 Lillian Place
Marina CA 93933-2218

12. Dr. David Finkleman 1
HQ USSPACECOM/AN
250 S Peterson Blvd STE 116
Peterson AFB CO 80914-3180

13. Edward Senasack 1
Naval Research Laboratory, Code 8200
Washington DC 20375

14. Manny Cohen 1
Aerospace Coporation
2350 El Segundo Boulevard
El Segundo CA 90245-4691

15. Dr. Sandra L. Scrivener, Code AA/SS 1
Naval Postgraduate School
Monterey CA 93943

16. Vice Admiral David Frost 1
USSPACECOM/UD
250 S. Peterson Ave Suite 116
Peterson AFB CO 80914-3010

17. Jerry Rhodes 1
TALON SHIELD
720 Erwin Avenue, STE 2306
Falcon AFB CO 80912-7200

18. Milo Whitson 1
Aerospace Coporation
PO Box 92957-M5/624
Los Angeles CA 90009-2957

19. Robert Erilane 1
Aerospace Coporation
PO Box 92957-M5/624
Los Angeles CA 90009-2957

20. Nicholas Gionis 1
Aerospace Coporation
PO Box 92957-M5/624
Los Angeles CA 90009-2957

21. Joseph Catuara 1
Aerospace Coporation
PO Box 92957-M5/624
Los Angeles CA 90009-2957

22. Albert Perrella 1
POET, STE 1100
1745 Jefferson Davis Highway
Arlington VA 22202

23. Gary Bartnick 1
POET, STE 1100
1745 Jefferson Davis Highway
Arlington VA 22202

24. John Shure 1
POET, STE 1100
1745 Jefferson Davis Highway
Arlington VA 22202

25. RADM Richard West 1
Ballistic Missile Defense Organization
Pentagon
Washington DC 20301-7100

26. COL Rich Ritter 1
Ballistic Missile Defense Organization
Pentagon
Washington DC 20301-7100

27. Henry Odom 1
Naval Surface Warfare Center
17320 Dahlgren Road
Dahlgren VA 22448

28. Ted Simms (K13) 1
Naval Surface Warfare Center
17320 Dahlgren Road
Dahlgren VA 22448

29. Jack Carr (K12) 1
Naval Surface Warfare Center
17320 Dahlgren Road
Dahlgren VA 22448

30. Robert Hill 1
Applied Research Laboratories
University of Texas at Austin
PO Box 8029
Austin TX 78713-8029

31. Andrew Goldfinger 1
Johns Hopkins University Applied Physics Laboratory
Johns Hopkins Road
Laurel MD 20723-6099

32. Fred Glaeser 1
Department of the Navy, CNO
N632, Room 5P773
Pentagon
Washington DC 20350-2000